Limited Labels for Unlimited Data: Active Learning for Speaker Recognition

Stephen H. Shum, Najim Dehak, James R. Glass

MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA, USA

{sshum, najim, glass}@csail.mit.edu

Abstract

In this paper, we attempt to quantify the amount of labeled data necessary to build a state-of-the-art speaker recognition system. We begin by using i-vectors and the cosine similarity metric to represent an unlabeled set of utterances, then obtain labels from a noiseless oracle in the form of pairwise queries. Finally, we use the resulting speaker clusters to train a PLDA scoring function, which is assessed on the 2010 NIST Speaker Recognition Evaluation. After presenting the initial results of an algorithm that sorts queries based on nearest-neighbor pairs, we develop techniques that further minimize the number of queries needed to obtain state-of-the-art performance. We show the generalizability of our methods in anecdotal fashion by applying our methods to two different distributions of utterances-per-speaker and, ultimately, find that the actual number of pairwise labels needed to obtain state-of-the-art results may be a mere fraction of the queries required to fully label the entire set of utterances.
Index Terms: speaker recognition, i-vectors, active learning

1. Introduction

Over the past 5 years, the i-vector approach has proven to be the best performing system as demonstrated in NIST speaker recognition evaluations (SRE) [1]. One of the keys to this success is a framework that easily allows the use of large amounts of previously collected and labeled audio to characterize and exploit speaker and channel variability. In the SRE scenario, data from thousands of speakers each making over 10 calls from at least 2 different handsets, collected in a consistent manner, has been readily available from previous years. However, it is unrealistic to expect a large set of labeled data from matched conditions when applying such a system to a new domain [2].

To that end, our previous work explored domain adaptation techniques that utilized, in an unsupervised manner, a set of matched, but unlabeled, data to supplement an existing system trained from labeled, but mismatched, data [2, 3]. Related work has also explored domain adaptation in the fully supervised sense [4]. While both scenarios are relevant, the actual deployment of a speaker recognition system into the real world is unlikely to warrant such extreme circumstances.

Obtaining a complete and exhaustive labeling of a set of N unlabeled utterances would require, in the worst case, $\frac{N \cdot (N-1)}{2}$ pairwise comparisons, but rather than deprive ourselves of any labels whatsoever as in the fully unsupervised case, perhaps we can obtain useful information from the expert labeling of some small fraction of these utterances. In this paper, we attempt to quantify how many labels are actually necessary to obtain state-of-the-art performance. We consider a scenario similar to the one considered in [2, 3, 4] in which we are provided with a vast quantity of matched, but completely unlabeled, data and are asked to build a speaker recognition system for subsequent audio.

To focus solely on the effect of limited labels, we remove the notion of mismatched domains – as in the domain adaptation problem previously explored – and assume that the unlabeled data at our disposal sufficiently matches the conditions in which we evaluate our speaker recognition system. Indeed, the existence of a previous system should only further reduce the number of expert labels required to obtain optimal performance, but we hope to keep things simple and not belabor this paper with the implementation details of previous work on domain adaptation. Finally, to incorporate expert labeling, we simply query an oracle for the answers to pairwise comparisons, such as, "Do utterances A and B contain the same speaker?"

2. Related Work

The setup of this problem moves us into the realm of semi-supervised and, more specifically, active learning, in which the system we build is allowed to ask for input and supervision on a limited number of specific examples [5]. We design our oracle setup to operate like humans might; in particular, a human asked to separate N utterances into homogeneous speaker clusters would likely break the problem down into a set of pairwise comparisons. This setup also provides a framework for a potentially crowd-supervised [6, 7] speaker recognition system, which we plan to pursue as future work.

For now, we consider a set of N unlabeled utterances and allow our system to ask for additional information in the form of pairwise comparisons. These comparisons yield a set of pairwise constraints that can be used to help in our process of active, semi-supervised clustering, which was formally developed into a variant of the general K-means algorithm in [8]. Our work utilizes a much more primitive connected-components algorithm that does not require an estimate of the number of clusters, K.

The movement towards requiring less labeled data to build a speaker recognition system has been explored in the past. The saga of work in [9, 10] managed to obtain solid results without the use of any labeled data. Based on previous work in speaker diarization, they utilize an unsupervised clustering technique resembling a K-means algorithm that also estimates the number of clusters via a heuristic that re-assigns the elements of small clusters to larger ones. As we continue to explore their methods, we reserve for future work a more in-depth consideration of unsupervised methods for speaker recognition. In this paper, we continue to allow the use of labels, albeit as few as possible.

The rest of this paper is organized as follows. Section 3 presents an overview of our system setup. Then Section 4 both outlines a naive initial algorithm that queries pairwise labels based on a nearest-neighbor approach and discusses its initial results. We propose techniques to further minimize the number of queries needed in Section 5, and finally, Section 6 concludes with a discussion of potential avenues for future work.

3. System Setup

We follow a similar setup to those presented in [2, 3, 4] and extract i-vectors [1] from a total variability matrix T of rank 600 and a gender-independent Universal Background Model containing 2048 Gaussian mixtures of acoustic features (MFCC+deltas). A more detailed background on how these hyper-parameters are obtained is beyond the scope of this paper but can be found in [1, 11, 12].

We use the data from the NIST Speaker Recognition Evaluations (SRE) of previous years (2004-2008) to train these hyper-parameters. These telephone calls contain roughly 3800 unique speakers (1100 male, 2700 female) and 33,000 phone calls. The average number of calls per speaker is roughly 8.7, and each speaker is represented by 2.8 different phone numbers. Our performance evaluation is conducted on the one conversation (1c) telephone data from condition 5 (normal vocal effort) of the SRE 2010 (SRE10) [13, 14].[1]

The training of these hyper-parameters does not require any labels; these initial i-vectors contain both speaker and channel (i.e., nuisance) information. Labels are not made available *a priori*, but if they were provided – or estimated via some clustering method – then we could obtain a within-class (WC) matrix, characterizing how i-vectors from a single speaker vary, and an across-class (AC) matrix, characterizing how i-vectors between different speakers vary [2]. The scoring function that has obtained state-of-the-art results is Probabilistic Linear Discriminant Analysis (PLDA) and is described in [15].

Our setup begins with a set of N utterances from the SRE data – the SRE10 data is *not* included here – represented as N i-vectors. We are allowed to ask some noiseless oracle for input in the form of pairwise labels; that is, "Are i-vectors i and j from the same speaker or different speakers?" The next section discusses a simple algorithm that makes use of these oracle queries to obtain respective WC and AC matrices, from which we can derive an appropriate PLDA scoring function and assess speaker recognition performance on SRE10.

4. Naively Labeling Nearest-Neighbor Pairs

In this section, we propose a naive algorithm based on querying nearest-neighbor pairs to a noiseless oracle to obtain speaker clusters from which we can obtain WC and AC matrices for PLDA training and subsequent evaluation.

4.1. The Algorithm

(a) Obtain pairwise cosine similarities in the form of an affinity matrix, A, where each entry A_{ij} represents the cosine similarity between i-vectors i and j. The cosine similarity (i.e., a length-normalized dot product) has been shown to be both a reasonable and fast metric for comparisons between i-vectors [1, 16, 17].

(b) Sort each row of A in descending order to obtain an ordered list of each node's nearest neighbors and their similarities. Specifically, row $\tilde{A}(i,:)$ is the sorted list of cosine similarity scores produced by i-vector i. Accompanying $\tilde{A}$ is a matrix $\tilde{I}$ such that $\tilde{I}(i,:)$ is the corresponding list of i-vector indices with whom i-vector i produced each of the scores in $\tilde{A}(i,:)$.

(c) The c^{th} column of $\tilde{I}$ and $\tilde{A}$ specifies the respective indices and scores for the c^{th} nearest neighbor of each of our N i-vectors. As $c = 1, 2, ...,$ query the pair $\left(i, \tilde{I}(i,c)\right)$ for each $i = 1, ..., N$. The number of unique pairs that are actually queried for each column $Q_c \leq N$, as each i-vector's ranking of its respective nearest neighbors will differ.[2] Operate on all i-vectors for a given column (i.e., c^{th} nearest neighbor) before moving on.

(d) Let G be an N-by-N binary matrix such that $G_{ij} = 1$ if i-vectors i and j originate from the same speaker, and $G_{ij} = 0$ otherwise. Initialize G as a matrix of all zeros, implying a completely disconnected graph, and when given a query that returns a same-speaker result, update G and its affected cliques. That is, if i-vectors i and j are a same-speaker pair (i.e., $G_{ij} = 1$), and $i \in I = \{i_1, i_2, ...\}$ while $j \in J = \{j_1, j_2, ...\}$ for cliques I and J, then this `clique-update` step automatically connects every element in I with every element in J. Because we assume that our oracle is noiseless, this is easy and saves us from making superfluous queries.

(e) As G becomes more and more connected as a result of querying the oracle, the size of the cliques in G will increase. Since these cliques correspond to perfectly pure speaker clusters, use the non-singletons to obtain WC and AC matrices for PLDA scoring. In the next section, we present our initial results as a function of the number of labels actively queried from the oracle.

In (c), the choice to operate on each column separately instead of simply querying the pairs corresponding to the highest global similarity scores, i.e., `global score sort`, is in an effort to maximize coverage of the i-vector space [18, 19]. In our unknown manifold, the highest similarity scores may simply correspond to parts of the manifold that are most densely packed. By requiring that each column be treated separately, we enforce a more `uniform coverage` of the i-vector space. This ensures that every node (i-vector) in our graph (dataset) is considered at least once every N queries. In the next section, the results of our initial experiments justify this hypothesis.

Along the lines of (d), we can further reduce the number of unnecessary queries by making use of information about different pairs that were previously encountered. For example, suppose the oracle returns (i, j) as a different-speaker pair, where $i \in X_q$ and $j \in Y_q$ for separate cliques X_q and Y_q in our graph G_q, where the subscript q denotes the state of the graph after q queries. Then for every subsequent query $q' > q$, if $x \in X_{q'}$ and $y \in Y_{q'}$, we already know without querying the oracle that (x, y) must be a different-speaker pair.

4.2. Initial Results

We visualize the results of our naive algorithm as a function of different subsets of the SRE data. Figure 1 shows the histograms of two different distributions of utterances-per-speaker in our sampled subset of SRE i-vectors. The subset represented in the plot at the top, `vanilla`, consists of all the utterances from 1000 speakers chosen uniformly at random, while the distribution displayed in the lower plot allows no more than five utterances per speaker, `max-5`.

[1] We also applied the methods discussed in this paper to various microphone conditions (1,2 – int-mic vs. int-mic; 4 – int-mic vs. room-mic) of the SRE10 and obtained trends similar to those reported on condition 5 (tel vs. tel) in this paper. In order to stay consistent with our previous work [2, 3, 4], we will continue reporting our performance based on the 1c telephone results from SRE10.

[2] That is, without loss of generality, i-vector j may be i-vector i's c^{th} nearest neighbor, but i may be j's $\hat{c}^{\text{th}}$ nearest neighbor, for some $\hat{c} \leq c$.

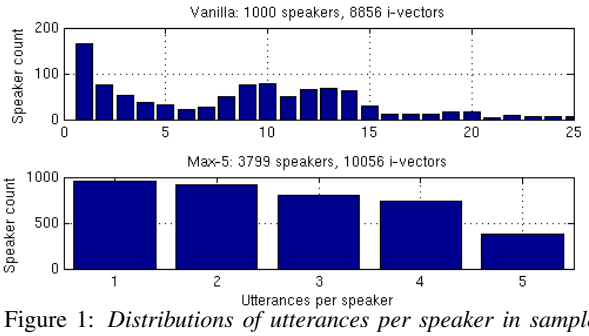

Figure 1: *Distributions of utterances per speaker in sampled subsets of SRE data: (top)* `vanilla` *– all utterances from 1000 randomly chosen speakers; (bottom)* `max-5` *– no more than five utterances from every speaker in the SRE data.*

It is helpful to consider how many queries are needed to verify a perfect cluster-labeling of all of the utterances in each of our SRE subsets considered. Note that we are not clustering from scratch, which would be $O(N^2)$, but simply verifying the validity of some hypothesized partition, $\mathcal{H}$, of N i-vectors from M speakers. This can be done using $(N - M) + \frac{M \cdot (M-1)}{2}$ queries. The first term, $(N - M)$, comes from verifying the purity of each cluster – i.e., a speaker cluster of size $|C|$ would require $(|C| - 1)$ queries to verify its purity – while the second term comes from verifying that each of the M clusters is indeed different from the rest. Verifying a clustering of the `vanilla` distribution requires 500,000 queries, while exhaustively checking a partition of the `max-5` distribution requires 7.2 million queries. We also show this information as part of the x-axis label at the bottom of Figures 2 and 3.

We plot our results as a function of the number of labels queried from the oracle (x-axis). While this includes queries of different-speaker pairs, this does not include the edges that are automatically created as a result of the `clique-update` step, only those in which the oracle is actively accessed. After every 1000 queries, we use the resulting cliques to define the speaker clusters, which are then applied to train a PLDA scoring function, and then run speaker recognition on the SRE10.

In Figure 2, we consider a number of different metrics. On the top plot, we show the number of different-speaker pairs encountered by the oracle (blue), the number of automatic connections made (green), and the resulting number of edges in our graph G (red). The middle plot compares the number of clusters found with the number of actual speakers, and the bottom plot shows both our subsequent results on SRE10 – in the form of an equal error rate (EER) – using the labels queried as well as the results using all possible labels. In these lower plots, we also justify our algorithmic choice in (c) of Section 4.1 by showing the difference in results obtained using our chosen `uniform coverage` approach (blue), which queries all pairs corresponding to each i-vector's respective K^{th} nearest neighbor before moving to the $(K+1)^{\text{th}}$, as well as the `global score sort` approach (green), which sorts *all* possible pairs in order of decreasing similarity score.[3] While both have similar asymptotic performance as expected, the `uniform coverage` approach obtains a better EER with fewer pairs queried.

A relatively small number of queries already produces results comparable to those obtained using all corpus labels. In fact, the `vanilla` distribution yields the same EER using

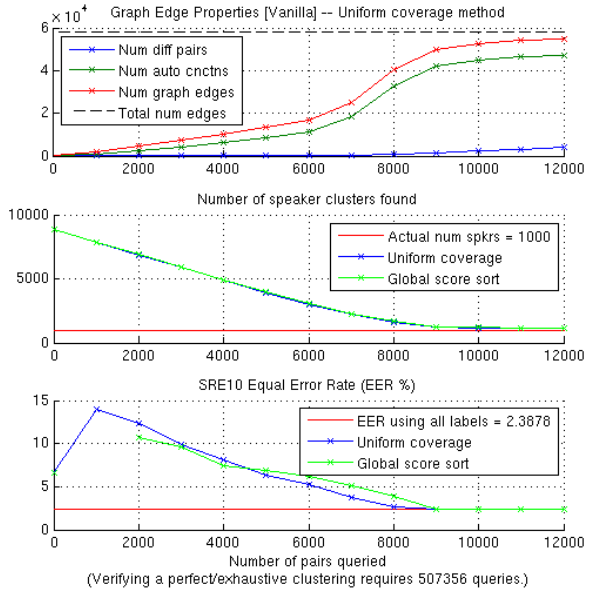

Figure 2: *Initial results obtained on the* `vanilla` *distribution of utterances per speaker.*

just about 9000 pairwise queries, which is far fewer than the 500,000 queries required to simply verify a perfect speaker clustering. On the other hand, using fewer than 5000 queries is worse than simply using the cosine similarity metric to evaluate on SRE10, which yields a 6.61% EER. This is because there are not enough edges on the graph to form reliable cliques that can faithfully model the WC and AC matrices for PLDA.[4] In light of this, detecting when we have enough queries to sufficiently represent our speaker space would be an interesting direction for future work. Conversely, the rate of change in the number of cliques detected (middle plot) may be a reasonable indicator for when we have utilized enough queries from our data.

The plot at the top of Figure 2 shows a rapid increase in the number of automatic connections made after 8000 queries. This is approximately where we begin querying pairs corresponding to second nearest neighbors. Once these connections are made, we start to see both the number of clusters found and the EER leveling off. The sudden spike in interconnected-ness makes sense given the `vanilla` distribution of utterances per speaker in our set of i-vectors; however, such a trend should not be expected for all such distributions, as can be seen for the `max-5` distribution in the top plot of Figure 3.

The two lower plots in Figure 3 show the rest of the results obtained on the `max-5` distribution, namely the number of clusters found (middle) as well as the resulting SRE10 EER (bottom) as a function of the number of pairs queried (x-axis). For now, note that the `uniform coverage` approach (blue) exhibits similar trends on both the `vanilla` and `max-5` distributions; the other approaches will be discussed in Section 5.

5. Refinements

So far, we have demonstrated the ability to obtain a good speaker recognition system using a mere fraction of the number of pairwise queries that would be required to label an entire corpus. Furthermore, these queries were chosen naively based on

[3] Furthermore, when pairs were queried in a completely random order, obtaining results comparable to those shown required an unreasonably large number of oracle queries.

[4] In fact, with just 1000 queries using `global score sort`, these covariance matrices were not even of sufficient rank to complete PLDA training; we chose not to pursue workarounds.

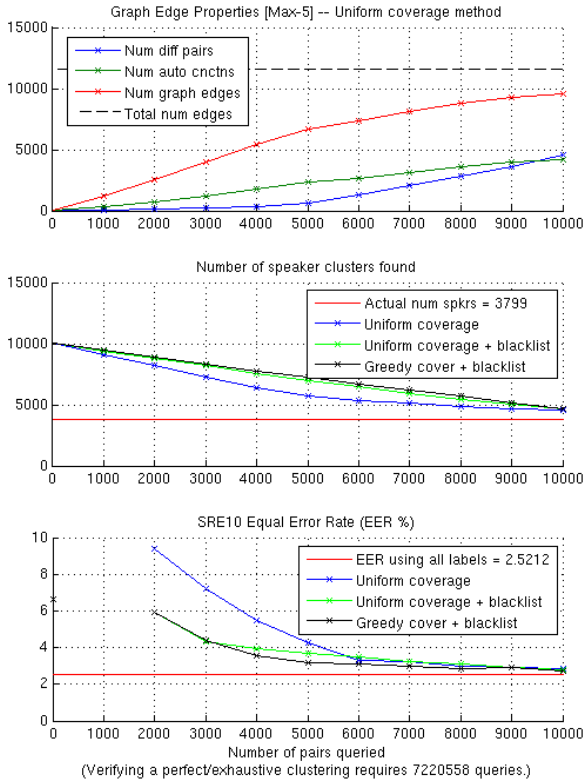

Figure 3: *Results obtained on the* `max-5` *distribution of utterances per speaker: (top) graph edge properties as a function of pairs queried; (middle & bottom) estimated number of speakers and resulting SRE10 EER for the* `uniform coverage` *approach as well as techniques discussed in Section 5.*

nearest neighbors according to a cosine distance metric. Having set an initial baseline, our interests turn to minimizing the number of active queries needed to obtain similar results.

5.1. Data Re-representation

One way to make use of the additional information obtained via pairwise queries would be to re-represent the data using a new pairwise affinity matrix A'. Instead of the cosine similarity, we could let A'_{ij} be the log-likelihood ratio (LLR) that i-vectors i and j belong to the same speaker; this can be computed via PLDA, whose AC and WC matrices can be determined by the cliques in our graph G_q after q queries. The mechanics of this are straightforward; the more interesting problem is determining when such a re-representation is appropriate. That is, we would like to know when the scoring function we are currently using to compare i-vectors – whether it is a cosine similarity or a PLDA LLR – is no longer suitable.

We consider a "blacklisting" approach, where each i-vector i is queried against its successively more distant neighbors, $\{j_1, j_2, j_3, ...\}$, until the oracle returns a different-speaker pair (i.e., $(i, j_d) \in \mathcal{D}$ for some $d \geq 1$). Once that occurs, we know that either our scoring function is no longer reliable or, ideally, that all the utterances involving the speaker in i-vector i have been found. As such, we add i to the blacklist and ignore subsequent comparisons involving it. This method greedily finds, with respect to the scoring function at hand, all of the same-speaker neighbors in the local neighborhood of i, thus accelerating the growth of speaker cliques. Once all – or some predetermined percentage – of the nodes have been blacklisted, we know that either all the speaker clusters in our data have been found or a re-representation of the data is necessary.

5.2. Greedy Manifold Sampling and Clique-Growing

Another potential way to reduce the number of queries is to be more selective about the order in which we pose them. For a given set of c^{th} nearest-neighbor pairs, our initial algorithm saw no difference between asking in a random order or in order of decreasing similarity score, but perhaps we can do better by modifying the "blacklisting" approach to also sample the entire i-vector space as quickly and uniformly as possible.

Suppose we start at node i and query its nearest neighbors, $\{j_1, j_2, j_3, ...\}$, in order of decreasing similarity until the oracle returns a different-speaker pair for (i, j_d). Then we pick a node k that is as far away (i.e., dissimilar) from $\{i, j_1, ..., j_d\}$ as possible.[5] As before, we query the neighbors of k until a different-speaker pair is returned, and so on. The hope is that this method will sample all corners of the manifold in as few queries as possible and, at the same time, grow as large of speaker clusters as possible with every node visitation.

The result of these refinements is shown in Figure 3, which compares the `uniform coverage` algorithm (blue) from Section 4.1 to the methods described previously. Without even needing to re-represent the data, the "blacklisting" approach (green) immediately yields a significant improvement on the SRE10 EER using just 2000 pairwise queries. This shows the effectiveness of greedily growing speaker cliques. Finally, the impact of the `greedy cover` approach (black) can be seen starting at 4000 queries, thus demonstrating that maximal coverage of the i-vector manifold can indeed help maximize performance in speaker recognition.

6. Conclusion

In this paper, we quantify the amount of labeled data needed to build a speaker recognition system. Beginning with unlabeled i-vectors and the cosine similarity metric, we query a noiseless oracle with nearest-neighbor pairs. Following promising initial results, we refine our techniques to maximize both cluster size and manifold coverage while minimizing both the number of queries needed and the resulting EER. Ultimately, we find that the actual number of pairwise labels needed to obtain state-of-the-art results is a mere fraction of the queries required to fully label an entire development set of utterances.

We have left open a number of avenues for future work. As a way to minimize the number of queries needed, we have not yet considered the idea of adding edges automatically and how to handle such potentially noisy labels; this may warrant the use of soft graph edge weights (i.e., $[0, 1]$) over hard assignments ($\{0, 1\}$). We should apply the developments of our previous work on domain adaptation and verify that knowledge gained from previously labeled data, albeit from a mismatched domain, can improve our initial representation of the data in the form of a better pairwise affinity matrix, A. Lastly, our work so far has been based on the existence of a noiseless oracle, but previous work has shown that both naive and expert human listeners can be imperfect [7]. In order to make our story more realistic, we plan to bridge the gap between our noiseless oracle and a crowd-sourced system for speaker recognition.

[5] This is easily done by averaging the corresponding rows of A and picking the i-vector corresponding to the minimum average similarity.

7. References

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[2] S. H. Shum, D. A. Reynolds, D. Garcia-Romero, and A. McCree, "Unsupervised clustering approaches for domain adaptation in speaker recognition systems," in *Proceedings of Odyssey: The Speaker and Language Recognition Workshop*, 2014.

[3] D. Garcia-Romero, A. McCree, S. Shum, N. Brummer, and C. Vaquero, "Unsupervised domain adaptation for i-vector speaker recognition," in *Proceedings of Odyssey: The Speaker and Language Recognition Workshop*, 2014.

[4] D. Garcia-Romero and A. McCree, "Supervised domain adaptation for i-vector based speaker recognition," in *Proceedings of ICASSP*, 2014.

[5] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.

[6] I. C. McGraw, "Crowd-supervised training of spoken language systems," Ph.D. dissertation, Massachusetts Institute of Technology, June 2012.

[7] W. Shen, J. Campbell, D. Straub, and R. Schwartz, "Assessing the speaker recognition performance of naive listeners using mechanical turk," in *Proceedings of ICASSP*, 2011.

[8] S. Basu, A. Banerjee, and R. J. Mooney, "Active semi-supervision for pairwise constrained clustering," in *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2004.

[9] H. Sun and B. Ma, "Unsupervised nap training data design for speaker recognition," in *Proceedings of Interspeech*, 2012.

[10] ——, "Improved unsupervised nap training dataset design for speaker recognition," in *Proceedings of Interspeech*, 2013.

[11] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 345–354, May 2005.

[12] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of inter-speaker variability in speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 5, pp. 980–988, July 2008.

[13] NIST, "Speaker recognition evaluation 2010," 2010, http://www.nist.gov/itl/iad/mig/sre10.cfm.

[14] A. Martin and C. Greenberg, "The 2010 nist speaker recognition evaluation (sre10)," 2010, http://www.nist.gov/itl/iad/mig/upload/SRE10_maineval_workshop_public_brief.pdf.

[15] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Proceedings of ICCV*, 2007.

[16] S. Shum, N. Dehak, and J. Glass, "On the use of spectral and iterative methods for speaker diarization," in *Proceedings of Interspeech*, 2012.

[17] O. Glembek, L. Burget, N. Dehak, N. Brummer, and P. Kenny, "Comparison of scoring methods used in speaker recognition with joint factor analysis," in *Proceedings of ICASSP*, 2009.

[18] Z. Karam and W. M. Campbell, "Graph embedding for speaker recognition," in *Proceedings of Interspeech*, 2010, pp. 2742–2745.

[19] Z. Karam, W. M. Campbell, and N. Dehak, "Graph relational features for speaker recognition and mining," in *IEEE Statistical Signal Processing Workshop*, 2011, pp. 525–528.