LARGE-SCALE SPEAKER SEARCH USING PLDA ON MISMATCHED CONDITIONS

Jeff Ma, Jan Silovsky, Man-hung Siu and Owen Kimball

Raytheon BBN Technologies, Cambridge, MA, USA

ABSTRACT

Recent work reported on fast speaker search over large speech data corpora has focused on using locality sensitive hashing (LSH) search with hashing functions approximating i-vector based cosine distances (CosDist) for model comparisons. Because of the superior performance of probabilistic linear discriminant analysis (PLDA) model reported on speaker identification (SID) in recent years, in this paper we focus on using PLDA for fast speaker search. It is challenging to approximate PLDA well with simple hashing functions, resulting in difficulty to combine it with LSH search. As an alternative, we adopt a clustering-based pruning strategy to speed up PLDA search. Our results show the strategy can significantly speed up search with minimal performance loss. Another focus of this work is on PLDA model adaptation to mismatched conditions under which the fast search runs. The technique we adopt to adapt the PLDA model is based on the LDA adaptation method reported in [1], primarily adapting the LDA transform. Our results show this adaptation improves PLDA performance significantly (over 25% relative) on data collected in different conditions. Our speed-up experiments running with adapted LDA show that gains from the adapted PLDA are retained after the speed-up.

Index Terms: speaker search, I-vectors, PLDA, cosine distance

1. INTRODUCTION

Speaker recognition techniques, including identification and verification, have advanced greatly in recent years. Two of them are i-vector features [2] and PLDA [3], which have resulted in superior performance in various speaker identification (SID) evaluations [4] [5]. Results have also been presented showing PLDA yields better performance than the CosDist metric [6][7]. Advances in communications and recording technologies have made large scale data collection with tens or hundreds of millions of hours of speech data per year possible for a single speech processing application. This has greatly increased the importance of fast speech processing algorithms. This paper presents work applying i-vector and PLDA techniques on speaker search for large speech corpora. Our first focus is on speeding up PLDA-based search while maintaining good performance. Often, the trained I-vector extractors and PLDA models are used to process data collected in different conditions, which usually causes performance degradation. Our second focus is to reduce degradation of the model when applied to new conditions.

2. RELATION TO PRIOR WORK

A large body of work on speaker search for large speech corpora has been reported. Recent work includes efforts reported in [8][9] [10] [11], which all use LSH [12] for fast search. In [8] the SID step is based on Gaussian mixture model (GMM) super-vectors. In [9] the authors employ factor analysis in their acoustic modeling step, but the model comparison does not use i-vectors. Since CosDist can be approximated well with LSH functions[13], both [10] and [11] run LSH search with hashing functions designed to approximate i-vector based CosDist. Both report good speed-ups with small losses in accuracy. We use i-vectors as well, but employ PLDA for model comparison. PLDA could not be approximated well with simple hashing functions, so it can not be easily combined with LSH search. For this reason we adopt a clustering-based pruning approach to prune the search space before running PLDA-based speaker search.

In [11], the authors chose YouTube data for investigating robustness of the search on mismatched conditions. However, they did not adapt the system to new conditions. We apply an unsupervised approach to adapt PLDA to new conditions based on the technique presented in [1].

3. PLDA FOR SID

3.1. I-vector features

Unlike joint factor analysis [14], [2] models the multiple variations as a single variable and the resulting formulation is

$$\mathbf{M}_i = m + \mathbf{T}v_i \tag{1}$$

where $\mathbf{M}_i$ is a supervector representing a speaker utterance, i, $\mathbf{T}$ is a low-rank rectangular matrix, m is the speaker- and channel-independent supervector, obtained with a universal background model (UBM). v_i is an i-vector carrying speaker variation information.

For SID, i-vectors are projected down to lower dimensional subspace features by using LDA and then normalized with WCCN and length normalization. These normalized features, y, are used to train PLDA models.

978-1-4673-6997-8/15/$31.00 ©2015 IEEE

3.2. PLDA

Given two feature vectors, y_1 and y_2, that are length normalized, the CosDist score is simply computed as the inner product, $< y_1 \cdot y_2 >$.

PLDA is more complex. A feature from a speaker can be modeled as [3],

$$y_{ij} = \mu + \mathbf{F}h_i + \mathbf{G}w_{ij} + \epsilon_{ij} \tag{2}$$

where y_{ij} denotes the j-th sample from speaker i, h_i variations between speakers; w_{ij} and ϵ_{ij} noises within individual speakers. μ represents the overall mean of all speakers. h_j and w_{ij} are assumed to be normal distributions. ϵ_{ij} follows a Gaussian with diagonal covariance Σ. All model parameters $\Theta = \mu, \mathbf{F}, \mathbf{G}, \Sigma$ are estimated on a given training data set.

Given two test features, y_1 and y_2, the PLDA score is computed as a log likelihood ratio, $log(P(y_1, y_2|\Theta)) - log(P(y_1|\Theta)) - log(P(y_2|\Theta))$, which can be derived to compute the following term,

$$-log|\mathbf{L_n}| + tr(\mathbf{L_n}^{-1} \mathcal{Z}_x \mathcal{Z}_x^T) \tag{3}$$

where,

$$\mathcal{Z}_x = \mathbf{F}^T(\mathbf{G}\mathbf{G}^T + \Sigma)^{-1} \sum_{i=1}^{n} (y_i - \mu) \tag{4}$$

$\mathbf{L_n}$ is a term varying only with n, the number of test features (in our case, $n = 2$), and the term $\mathbf{F}^T(\mathbf{G}\mathbf{G}^T + \Sigma)^{-1}$ can be pre-computed.

According to [17], the fastest algorithm for multiplication, inversion and determinant matrix (DxD) all have computational complexity $O(D^{2.73})$. So, the complexity for computing PLDA score for one trial is approximately $O(2 \cdot D^{2.73} + D^2)$, with ignoring lower-order computations. Meanwhile, computation complexity for CosDist is $O(D)$. Hence, computing PLDA scores is roughly $2 \cdot D^{1.73} + D$ times slower than computing CosDist scores.

3.3. LDA adaptation

We employ within-class correction (WCC) technique in [1] to adapt PLDA, because its unsupervised characteristics, requiring no labeling of adaptation data, suits our situation well. The WCC aims at suppressing variability in a direction of the shift between modes corresponding to the training and adaptation data that does not carry any useful information for differentiating speakers. To identify sources in adaptation data, we employ the same clustering approach as described in [1]. The resulting LDA transform is then estimated in the standard way using the between-speaker covariance $\mathbf{\Sigma}_B^{(trn)}$ and corrected within-speaker covariance given by

$$\mathbf{\Sigma}_W^{(corr)} = \mathbf{\Sigma}_W^{(trn)} + \alpha \mathbf{\Sigma}_{BSrc}^{(apt)} \tag{5}$$

where α is the mixing weight (we set it to 1.0) and $\mathbf{\Sigma}_{BSrc}^{(apt)}$ is the correction term estimated as shown in [1].

With LDA adapted we re-train the down-stream models, WCCN and PLDA, for adapting them to the new condition.

4. SPEED-UP ALGORITHMS

As shown in Section 3.2, PLDA is more complex than CosDist. We feel it is a great challenge to find simple hashing functions to approximate PLDA, resulting in difficulty of combining PLDA with LSH search. Instead, we adopt a straightforward approach: cluster audios in the corpora into M classes based on a distance metric; for a given query audio, search the class closest to it; score the given audio against samples only from the closest class. More generally, the top-n closest classes, $n > 1$, can be selected to balance between speed-up and loss in search accuracy. Consequently, audio samples from the top-n classes are scored.

As described in 3.1, one vector feature, y, is extracted for each audio file. These features are used to carry out the clustering and search. We investigate two commonly used clustering methods , the Gaussian mixture model (GMM) and K-means methods. Our GMM-clustering procedure is: starting with one Gaussian, estimated from all data, iteratively split the Gaussians until the required number is reached. At each split we run 5 EM iterations to update the Gaussians. After the GMM is estimated, we treat each Gaussian mixture as one class, represented with the mixture mean and variance. We found that using the Gaussian posteriors to find the top-n classes was not good because the posteriors were either 1 or 0 for a majority of cases and thus resulted in only one class selected even if more than one is desired. So we changed to select top-n classes based on CosDist to the means of the classes. Our results showed that the CosDist-based selection was significantly better than the posteriors-based. In all experiments we used the CosDist-based selection.

We run K-means clustering with CosDist metric as follows: randomly select M samples as the class centroids; assign samples to the classes based on their cosine distances to the class centroids; update each class centroid with the mean of samples assigned to the class; repeat the assignment and update a number of times (5 in our experiments). After the classes are estimated, top-n classes are selected based on CosDist between the test sample to the class centroids.

As noted, finding top-n classes introduces extra search time, which needs to compute M CosDist scores and thus the complexity is $O(MD)$ for each query. Suppose each class includes a samples on average, complexity for computing PLDA scores of the samples in the top-n classes is approximately $O(na(2D^{2.73} + D))$. So the ratio of the extra search time over the PLDA scoring time is $M/(na(2D^{1.73} + D))$. In general, D is in the range $[100, 200]$ and makes the ratio a small value, so the extra search time is negligible, compared to the PLDA scoring time. [1] Therefore, we simply measure speed-up as the reduction degree of the trial set, computed as

[1] However, it would not be negligible if we used CosDist metric to score the samples as done in [10] [11]. The top-n selection can be sped up with advanced algorithms, like the binary-tree methods. Since it was not a major focus, we did not pursue this further.

the number of original trials divided by the number of trials after pruning.

5. EXPERIMENTS

5.1. Data

We chose data from three different data collections released by the LDC. The first is the conversational telephone Switchboard data collections, including both Switchboard-1 and SwitchBorad-2 releases (SWBD1n2). The second comes from the NIST SRE data corpus. We used only the telephone data used in the SRE evaluations from years 2004 to 2008 (SRE04-08). The third comes from the DARPA RATS (robust automatic transcription of speech) SID data releases. We selected only the clean and channel G data of the releases for the SID task (RATS-sG). The "SWBD12" and "SRE04-08" are narrow-band data (8K sampling) and the "RATS-sG" wide-band data (16K sampling). In our experiments we up-sampled the narrow-band data to 16K for the convenience of using our tools for building the RATS evaluation systems. The three sets include 33K, 36K and 29K audio files, respectively. 845, 1,577 and 1,336 hours of speech data were obtained for the three sets, respectively, based on the speech activity detection system we developed for RATS [18].

We used the "SWBD1n2" data for training and the other two sets for test. Since the RATS data was collected under significantly different conditions, there is a mismatch between training and test data, which we attempt to improve using adaptation techniques. From the "SRE04-08" data we chose 28,500 telephone audio files as enrollment samples and 600 files as test samples. From the "RATS-sG" data we chose 31,497 audio files as enrollments and 400 as tests. Merging the samples from these two data sets resulted in 59,997 enrollment and 1,000 test samples. Then we created a trial set by including all the enrollment and test sample pairs. By design, the 1,000 test samples were randomly selected from 1,000 different speakers who have at least one different audio samples in the enrollment samples. In this way, it was guaranteed that each test sample had at least one target trial. The total number of trials is 60M, out of which 6,157 were targets. We use "sre-rats" to denote this trial set. We measure SID performance in terms of equal error rate (EER), treating missing and false alarm errors equally. All EERs are computed on this trial set.

5.2. Baseline SID performance

We trained a 600-dimension I-vector extractor with a UBM having 2,048 mixture components, then estimated a 100-dimension LDA transform and WCCN(100x100) to transform and normalize the I-vectors, and finally trained a PLDA model with the normalized 100-dimensional features. The EER of this PLDA model measured on the "sre-rats" trial set is 6.11%.

Then, using the approach described in Section 3.3 we adapted the LDA with the "sre-rats" data. With the adapted LDA we re-trained the WCCN matrix and PLDA model. This adapted PLDA produced an EER, 3.88%. So the adaptation reduced the EER by 26% relative. The performance shown here serves as baseline for the speed-up experiments we report later.

5.3. Clustering-based speed-up

Apparently, different combinations of the M and n values can result in the same speed-ups. We first ran the GMM-clustering on the 60K enrollment samples three times, generating 1,024, 2,048 and 2,560 classes, respectively. For each of the three clustering we used different top-n values to select trials (or to prune trials). Second, we ran the K-means clustering on the data twice, generating 1,024 and 2,560 classes. Similarly, we conducted pruning with different top-n values on the two clustering.

To fairly compare different speed-ups we always measured EER on the whole "sre-rats" trial set, by counting pruned target trials as misses. We measured EERs at the different speed-ups and then plotted the EERs over the speed-ups to demonstrate trends of EER changes over speed-ups. Plots for the 3 GMM-clustering and the 2 K-means clustering are shown in Figure 1.

Comparing the 3 GMM clustering, we see that at the same speed-up the use of a larger number of classes resulted in less EER degradation. With the use of 2,560 GMM classes, the average number of samples in each class is 23.4. In our experience, less than 20 samples would not give a reliable Gaussian estimate, so we did not investigate more than 2,560 classes. Comparing the two K-means clustering, we have the same observation.

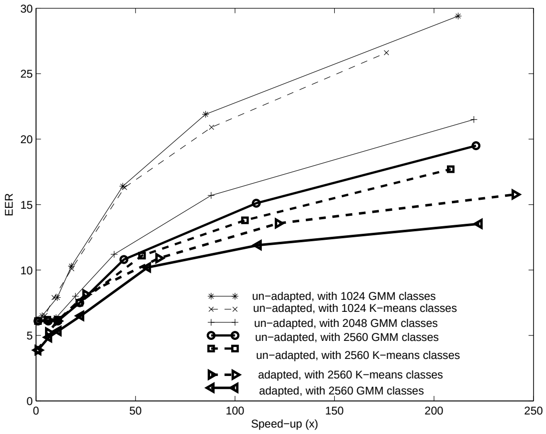
Comparison of the GMM and K-means clustering with 2,560 classes shows that use of the K-means classes caused less EER degradation at the same speed-up. Therefore, the use of 2,560 K-means classes performed the best in this scenario. According to the formula given in Section 4, with $M = 2,560$ classes, the top-n class selection only takes 3.7% of the PLDA scoring time, even at the most aggressive pruning (selecting only top-1 class), which is a small cost as noted before.

5.4. Speed-up with LDA adaptation

With the adapted LDA we re-ran both GMM-clustering and K-means clustering to generate 2,560 classes, and then ran different pruning and measured the EER at each pruning. For this scenario the EERs were measured with the adapted PLDA. The plots are also shown in Figure 1.

First, comparing the adapted GMM and K-means clustering, we see that the GMM-clustering out-performed the K-means clustering. This is the opposite of the behavior observed in the un-adapted scenario. The reason could be

Fig. 1. Performance (EER) vs Speed-ups with the GMM and K-means clustering methods



that the adapted LDA reduced within-speaker covariances and thus helped the GMM-clustering more due to use of covariances in the GMM-clustering but not in the K-means clustering.

Comparing the adapted 2,560 GMM clustering against its un-adapted counterpart, we can see that the PLDA adaptation gain (with no trial pruning) is roughly retained at different speed-ups. For the adapted 2,560 K-means clustering, although it produced better performance than its un-adapted counterpart, the PLDA adaptation gain shrunk with speed-ups, especially small speed ups (less than 50x). This indicates that it is better to use the GMM-clustering in scenarios where PLDA is adapted.

It can be observed that in the un-adapted scenario the use of either the GMM or the K-means 2,560 classes cause no EER degradation at small speed-ups in the range of $[1, 10]$. On the contrary, in the adapted scenario with 2,560 classes, there is degradation in the same range. We found the reason was the use of CosDist during the clustering (for the top-n class selection). The EER of the CosDist model in the un-adapted case was 5.37%, which is better than the EER 6.11% of the un-adapted PLDA model, so CosDist metric helped prune impostors on which the PLDA made errors. After adaptation, the EER of the CosDist was 3.84%, similar to the 3.88% EER of the PLDA model, thus CosDist metric no longer provided any helps in trial pruning.

The baseline EERs (denoted as "0x") and EERs computed at three different speed-up values, 11x, 110x and 1100x, with the 2,560 classes in both the un-adapted and adapted scenarios are shown in Table 1, where "un-adapt" denotes the un-adapted scenario, "adapt" the adapted scenario and "adapt-m" a mixed scenario where speed-up runs with the adapted PLDA and the un-adapted 2,560 GMM classes. First, it can be seen that in the un-adapted scenario there is no EER degradation even at speed-up equal to 11x. Second, comparing "adapt-m" and "adapt", we see that re-clustering with adapted LDA is not necessary when pruning is less aggressive (like speed-up $<$ 11x), but the re-clustering is desirable when speed-up becomes more aggressive (like $>$ 100x), because it produces significantly lower EERs.

Table 1. *EER at different speed-up with GMM and K-means clustering*

clustering	0x	11x	110x	1,100x
un-adapt, KM2560	6.11	6.17	13.77	26.31
un-adapt, GMM2560	6.11	6.09	15.12	28.24
adapt-m, GMM2560	3.88	5.37	15.12	28.24
adapt, GMM2560	3.88	5.34	11.60	24.48

Table 2. *Comparison (on EER) of the LSH approach versus the clustering approach*

Percentage (speed-up)	RDLSH	un-adapt, KM2560
0.01 (100x)	31.59	13.77
0.05 (20x)	12.97	7.49

5.5. Comparison to the LSH approach

We tried the random projection LSH (RDLSH) fast search approach reported in [10] on the trial set used here. [2] Since in [10] the CosDist metric was used for scoring the trials, we also used the CosDist to score the trials selected with the "un-adapt, KM2560" clustering. EERs measured at the same pruning rates from uses of both approaches are listed in Table 2, where "Percentage" indicates the percentage of trials that survive the pruning. As can be seen, the clustering-based approach had much lower EERs at the same pruning rates.

6. CONCLUSIONS

We have explored the use of clustering-based pruning for speeding up the speaker search using PLDA on mismatched conditions, with training from Switchboard data and test from the SRE and RATS data sets. Our results show that in the scenario without model adaptation K-means clustering performed better and was able to prune more than 90% of the search space without EER degradation. In the scenario with model adaptation, GMM-clustering was better and was able to prune 90% of the search space with minimal EER degradation. We have also investigated adapting PLDA to data from new conditions. Our results show that the adaptation method we employed produced gains of more than 25% relative while adapting from Swithboard to SRE and RATS data. Our experiments also show that with the adapted clustering, the PLDA adaptation gains are retained at different speed-ups. Hence, clustering-based pruning approaches are very efficient in speeding up PLDA-based speaker search.

[2] Many thanks to our colleague, Ryan Leary, for running this

7. REFERENCES

[1] O. GLEMBEK, J. MA, P. Matejka, B. ZHANG, O. PLCHOT, L. BURGET, “Domain Adaptation Via Within-class Covariance Correction in I-Vector Based Speaker Recognition Systems”, in *Proceedings of ICASSP 2014*, pp. 4060-4064. ISBN 978-1-4799-2892-7.

[2] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” in *IEEE Transactions on audio, speech and language processing*, vol. 19, no. 4, 2011.

[3] Simon J.D. Prince and James H. Elder, “Probailistic Linear Discriminant Analysis for Inferences about Identify”, in *IEEE 11th Internatinal conference on Computer Vision, 2007.*

[4] L. Ferrer, M. Mclean, N. Scheffer, Y. Lei, M. Graciarena and V. Mitra, “A Noise-Robust System for NIST 2012 Speaker Recognition”, in *Proc. of ICASSP 2013*.

[5] O. Plchot, S. Matsoukas, P. Matejka, etc. “Developing a Speaker Identification System for the DARPA RATS Project”, in *Proc. of ICASSP 2013*.

[6] P. Matejka, O. Glembek, F. Castaldo, M. J. Alam, O. Plchot, etc, “Full-Coverance UBM and Heavy-Tailed PLDA in I-vector Speaker Verification”, in *Proc. ICASSP 2011*.

[7] M. Senoussaouii, P. Kenny, P. Dumouchel and N. Dehak, “New Cosine Similarity Scorings to Implement Gender-independent Speaker Verification “, in *Proc. of ICASSP 2013*

[8] M. A. Pathak and B. Raj, “Privacy-preserving speaker verification as password matching”, in *Proc. of ICASSP 2012*.

[9] W. Jeon and Y. Cheng, “Efficient speaker search over large populations using kernelized locality-sensitive hashing”, in *Proc. of ICASSP 2012*.

[10] R. Leary and W. Andrews, “Random Projections for Large-Scale Speaker Search”, in *Proc. of Interspeech 2014*.

[11] L. Schmidt, M. Sharifi, and L. Moreno, “Large-scale speaker identification”, in *Proc. of ICASSP 2014*.

[12] S. Har-Peled, P. Indyk, and R. Motwani, “Approximate nearest neighbor: Towards removing the curse of dimensionality”, in *Theory of Computing*, vol. 8, no. 14, 2012.

[13] M. Charikar, “Similarity estimation techniques from rounding algorithms”, in *ACM Symposium on Theory of Computing (STOC)*, 2002.

[14] P. Kenny, G. Boulianne, P. Ouellet and P. Dumouchel, “Joint factor analysis versus eigenchannels in speaker recognition”, in *IEEE Trans. on Audio, Speech, Language Process*, 2007, Vol. 15, No. 4

[15] D. A. Reynolds, T. F. Quatieri and R. B. Dunn, “Speaker Verification Using Adapted Gaussian Mixture Models”, in *Digital Signal Processing*, 2000, Vol.19, No.41

[16] W. M. Campbell, D. E. Sturim, D. A. Reynolds and A. Solomonoff, “SVM based speaker verification using a GMM-supervector kernel and NAP variability compensation”, in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, 2006.

[17] V. V. Williams, “Breaking the Coppersmith-Winograd Barrier”, 2001.

[18] T. Ng, B. Zhang, L. Nguyen, S. Matsoukas, K. Vesely, P. Matejka, X. Zhu, and N. Mesgarani, “Developing a speech activity detection system for the darpa rats program”, in *Proc. of Interspeech 2012*.