INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

The Opensesame NIST 2016 Speaker Recognition Evaluation System

Gang Liu, Qi Qian, Zhibin Wang, Qingen Zhao, Tianzhou Wang, Hao Li, Jian Xue, Shenghuo Zhu, Rong Jin, Tuo Zhao[1*]

Alibaba Group (U.S.) Inc.
[*]University of Missouri, U.S.

{g.liu, qi.qian, zhibin.waz, qingen.zqe, konghai.wtz, lihao.hl, jian.xue, shenghuo.zhu, jinrong.jr}@alibaba-inc.com, tz579@mail.missouri.edu

Abstract

Last two decades have witnessed a significant progress in speaker recognition, as evidenced by the improving performance in the speaker recognition evaluations (SRE) hosted by NIST. Despite the progress, only a few research is focused on speaker recognition with short duration and language mismatch condition, which often leads to poor recognition performance. In NIST SRE2016, these concerns were first systematically investigated by the speaker recognition community. In this study, we address these challenges from the viewpoint of feature extraction and modeling. In particular, we improve the robustness of features by combining GMM and DNN based iVector extraction approaches, and improve the reliability of the back-end model by exploiting symmetric SVM that can effectively leverage the unlabeled data. Finally, we introduce distance metric learning to improve the generalization capacity of the development data that is usually in limited size. Then a fusion strategy is adopted to collectively boost the performance. The effectiveness of the proposed scheme for speaker recognition is demonstrated on SRE2016 evaluation data: compared with DNN-iVector PLDA baseline system, our method yields 25.6% relative improvement in terms of min_Cprimary.

Index Terms: symmetric SVM, distance metric learning, SRE2016, language mismatch, speaker recognition

1. Introduction

In NIST Speaker Recognition Evaluation (SRE), the major task is speaker detection, that is, to determine whether a specified target speaker is speaking during a given speech segment. Like the past sessions, SRE2016 focuses on telephone speech records, but it also has some new challenges [1]. For example, test segments have more duration variability than in previous evaluations, which leads to the fact that test durations are evenly distributed among 10s~60s; the enrollment and test data were collected outside North America, from which severe language mismatch is expected, given the fact that most of the assistive development data were collected in North America. We approach these issues via better iVector extraction and modeling approaches.

To be specific, we will introduce the experiment corpus on Sec. 2, iVector extraction methods in Sec. 3, and modeling approach in Sec. 4. Different system configurations are explained in Sec. 5. Experiment results are documented and analyzed in Sec. 6. Timing and memory usage are reported in Sec. 7.

2. Experiment corpus

All the data used in modeling comes from NIST SRE2016 under LDC data license agreement. NIST SRE2010 data is used in the early stage of system verification. In the later stage, Development set of Call My Net is used for system performance tuning and fusing. Other than the data used for enrollment and verification, we also need data to assist system modeling, which is detailed in Table 1. Note that "Call My Net" is the 2472-session based unlabeled audio data (i.e., no speaker id, gender, language), which includes 2272 sessions for the major language and 200 ones for the minor language [1].

Table 1: *Data partition for system modeling. SRE04~08 includes NIST SRE2004, SRE2006, SRE2008. "Fisher EN" is the English speech and text data from Fisher corpus. "X" means "being chosen".*

Data / Usage	Switch -board	SRE 04~08	Fisher EN	Call My Net
UBM	X	X		
iVector Extractor	X	X		
LDA, PLDA, DML		X		
DNN			X	
SVM imposter				X

3. iVector extraction

Three approaches are adopted for iVector extraction. They are detailed as follows.

3.1. GMM-iVector extraction

The front-end consists of 20 MFCCs with a 25ms frame-length. The features are mean-normalized over a 3s window. Delta and double delta are appended to create 60 dimensional frame-level feature vectors. The non-speech frames are then eliminated using energy-based voice activity detection (VAD) [2]. Then a GMM (i.e., UBM) with 2048 mixture components is trained with the data specified in Table 1. iVector extractor is trained with the same data as for the UBM. For iVector extraction, two dimension sizes are investigated: 600 dimension (600D) and 800 dimension (800D).

[1] The work is done while the author was an intern at Alibaba Group.

3.2. DNN-iVector extraction

The input for DNN system are features equivalent to filterbanks: 40 MFCCs without cepstral truncation. The frame-length is 25ms. Time delay deep neural network (TDNN) [2] is used to replace the UBM as in Sec. 3.1. The TDNN has six layers. The softmax output layer computes posteriors for 5297 triphone states, which are used to train an iVector extractor. 600 dimension iVectors are extracted.

3.3. LSTM-iVector extraction

Same as DNN-iVector in Sec. 3.2, a 3-hidden-layer-LSTM-HMM-based [3] acoustic model is trained, in which each hidden layer contains 1024 memory cells with 512 projection units and peephole connections. This model is optimized by truncated Backpropagation Through Time (BPTT) [4][19]. The truncated chunk size is set to be 20 and output predictions are delayed for 5 frames. Each mini-batch has 128 truncated chunks.

4. Backend modeling

All the iVectors in this study are processed with length normalization [5]. For dimension reduction, LDA and distance metric learning are investigated. We skip LDA's explanation here since it is well-known. We begin with PLDA, then propose two back-ends: symmetric SVM and distance metric learning (DML).

4.1. Probabilistic Liner Discriminative Analysis (PLDA)

For PLDA modeling, iVector is post-processed with mean subtraction and length normalization, followed by LDA dimension reduction and scoring. To derive optimal performance, we need in-domain data. We can use unlabeled data from Call My Net (Table 1). However, we do not have labels, and thus we only use it to derive mean and do a global mean subtraction on all the iVectors. The between-class and within-class covariance matrices of the PLDA backend are estimated from the dataset described in Sec. 2.

4.2. Symmetric SVM

It is shown that SVM-based back-end can boost the speaker recognition performance [6, 7, 8, 9, 10, 11]. Similar framework is followed here with some modification. It is known that each trial involves an enrollment speaker and a test segment. Using iVectors from DNN-iVector PLDA framework as data inputs, symmetric SVM can be illustrated as Fig. 1. Mathematically, two SVMs are built for each trial in Symmetric SVM as in Eq. (1):

$$f(x_{tst} \mid SymmetricSVM(X^{i}_{enr}, X_{imp})) = \frac{1}{2}\left(f(x_{tst} \mid SVM_{pos}(X^{i}_{enr}, X_{imp})) + f(X^{i}_{enr} \mid SVM_{neg}(x_{tst}, X_{imp})) \right) \quad (1)$$

where in the first SVM (or positive SVM, SVM_{pos}), the mean of all the data samples from the involved ith enrollment speaker is used as positive sample, and all the iVectors of the unlabeled data are used as negative samples. In the second SVM (or negative SVM, SVM_{neg}), the iVector of the test segment is used as positive sample, and all the iVectors of the unlabeled data are used as negative samples. Linear kernel is adopted. The deployment is based on LibSVM [12]. The output probability is converted into log probability with logarithm operation.

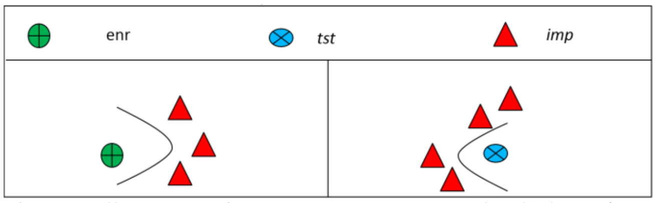

Figure 1: *Illustration of symmetric SVM. Top panel is the legend for enrollment (enr), test (tst) and imposter (imp) data, respectively. The bottom left and right panels are for positive SVM (SVM_{pos}) and negative SVM (SVM_{neg}), respectively.*

4.3. Distance metric learning (DML)

Distance metric learning (DML), which aims to push the data from the same class together and pull away the data from different classes, has been successfully applied for many real world applications, e.g., image classification [13], face verification [14], etc. Recently, DML has attracted the attention from the speaker verification community and shown promising results [15][16].

Given the dataset $X \in R^{n\times d}$, DML tries to learn a good distance metric $M \in S_{+}^{d\times d}$, such that

$$\forall x_i, x_j, x_k, x_l, dist(x_i, x_j) - dist(x_k, x_l) < \delta \quad (2)$$

where δ is the margin to separate the pairs from the same class from those with different labels. x_i and x_j share the same label while x_k and x_l have different labels.

Unlike the conventional DML, we use cosine similarity in this work as suggested in [16] and the target becomes

$$\forall x_i, x_j, x_k, x_l, S(i, j) - S(k, l) > \delta \quad (3)$$

where

$$S(i, j) = x_i^T L L^T x_j / (\| x_i^T L \| \| x_j^T L \|) \quad (4)$$

The matrix $L \in R^{d\times d}$ can be learned by solving the following optimization problem:

$$\min_{L\in R^{d\times d}} \sum \ell(x_i, x_j, x_k, x_l; L) + \frac{\lambda}{2} \| L - I \|_F^2 \quad (5)$$

where the former item is the loss function and the latter is the regularizer to avoid overfitting. I denotes the identity matrix. In this work, the logistic loss is applied for its property of smoothness:

$$\ell(x_i, x_j, x_k, x_l; L) = \log(\exp(S(k, l) - S(i, j) + \delta) + 1) \quad (6)$$

Given the large number of constraints (i.e., up to $O(n^4)$), we solve the problem by using stochastic gradient descent (SGD) with the mini-batch updating strategy [17].

Finally, to explore the information from unlabeled in-domain data (CallMyNet), we apply PCA without dimension reduction to the training data as pre-process while the projection matrix of PCA is learned from unlabeled data.

5. System configuration

For the abovementioned front-end and back-ends, there are many combinations. We propose seven sub-systems by rule of thumb. Then we diversify final submission with different fusion strategy.

5.1. Sub-systems

Among different front-end and back-end combinations based on Sec. 3 and 4, we finalized the following seven configurations:

- Sub-system 1: DNN-iVector PLDA 600D. The front-end is DNN based 600-Dimension iVector and the back-end is PLDA, which is also taken as our baseline system.
- Sub-system 2: DNN-iVector Symmetric SVM 600D. The front-end is DNN based 600-Dimension iVector (same as in sub-system 1). The back-end is Symmetric SVM.
- Sub-system 3: GMM-iVector PLDA 600D. The front-end is GMM based 600-Dimension iVector. The back-end is PLDA.
- Sub-system 4: GMM-iVector PLDA 800D. The front-end is GMM based 800-Dimension iVector. The back-end is PLDA.
- Sub-system 5: DNN+GMM-iVector DML 1200D. The front-end is the concatenation iVector from sub-system 1 and 3. The back-end is Cosine Similarity-based distance metric learning.
- Sub-system 6: DNN-iVector DML 600D. The front-end is DNN based 600-Dimension iVector (same as in sub-system 1). The back-end is Cosine Similarity-based distance metric learning.
- Sub-system 7: GMM-iVector DML 600D. The front-end is GMM based 600-Dimension iVector (same as in sub-system 3). The back-end is Cosine Similarity-based distance metric learning.

The last three sub-systems are variations with different feature inputs. The performances of these seven sub-systems are detailed in Table 3. Since we did not observe any gain from LSTM-iVector extraction, we skip its result.

5.2. Fused system and final submission

To make full use of the diversity of the abovementioned sub-systems, Bosaris toolkit is used for fusion [18]. Both linear and pool adjacent violators (PAV) calibration are explored on the scores. After the calibration, a linear fusion parameter set is learned from development data set and applied to evaluation set.

Overall, four fused systems are submitted:

- Submission 1: fixed-primary. This is the fusion of sub-system 1 through 7 with linear calibration.
- Submission 2: fixed-contrastive 1. This is the fusion of sub-system 1 through 4 with linear calibration.
- Submission 3: fixed-contrastive 2. This is the fusion of sub-system 1 through 7 plus quality measure from duration, etc. with linear calibration.
- Submission 4: Open-primary. This is the fusion of sub-system 1 through 7 with PAV calibration. Different from the first three submissions, this is for open condition.

Their performances on both development and evaluation data for fixed and open condition are summarized in Table 4. The fusion parameter is tuned on development data. The lackluster performance of submission 4 on evaluation data suggests that an extra set of data is needed for better generation.

6. Results and analysis

6.1. Pilot Results

In the early stage of system development, the effort is focused on the first four sub-systems introduced in Sec. 5.1. Performances of sub-system are detailed in Table 2. The scoring is prepared with SRE2016 official scoring script, which provides three metrics: EER, min_Cprimary, and act_Cprimary under two categories: "equalized" and "unequalized" [1]. Note, min_Cprimary and act_Cprimary is normalized minimum detection cost function and normalized detection cost function, respectively. Without calibration, all the scores of the four systems lead to meaningless act_Cprimary (the calibrated ones are put within parenthesis). This suggests the importance of an effective score calibration.

Table 2: *Pilot exploration of sub-system performance on Development set of CallMyNet corpus. For "Equalized" act_Cprimary, calibrated metric is put within parenthesis.*

System	Metric	Equalized	Unequalized
Sub-system 1: DNN-iVector PLDA	EER	17.19	18.12
	min_Cprimary	0.8097	0.7953
	act_Cprimary	77 (0.8217)	95
Sub-system 2: DNN-iVector Symmetric SVM	EER	18.26	19.16
	min_Cprimary	0.7243	0.7201
	act_Cprimary	1 (0.7599)	1
Sub-system 3: GMM-iVector PLDA 600D	EER	20.86	19.14
	min_Cprimary	0.7967	0.7759
	act_Cprimary	49 (0.8438)	65
Sub-system 4: GMM-iVector PLDA 800D	EER	21.58	19.42
	min_Cprimary	0.7666	0.7431
	act_Cprimary	49 (0.8297)	65

6.2. Subsystem Results

The sub-systems' performance are detailed in Table 3. Given the EER's limitation resulting from single operation point, we rely on Cprimary as evaluation metric.

Our first observation is, in terms of performance of individual system, sub-system 2 (DNN-iVector Symmetric SVM) is the most competitive. It offers consistent performance on both development and evaluation data sets. The reason for this is that Symmetric SVM can build reliable hyperplane even without knowing the label information of the imposter data. For PLDA, to leverage the benefit of the in-domain data, clustering technology can be employed. Our initial investigation shows that the uncertain classification noise overrides the gain. Instead, we use global mean subtraction mentioned in Sec. 4.1 to partially mitigate the impact of the language mismatch, then we use labeled out-of-domain data to derive the PLDA model.

The second observation is that the feature concatenation based fusion (sub-system 5) can effectively boost performance. Unlike past NIST SREs such as SRE2010, DNN-based iVector does not demonstrate significant improvement against GMM-based iVector. This can be explained by the fact that the DNN component is trained with English language, while the enrollment and test data are non-English. This set-up does not produce effective triphone states as in language matched

Table 3: *Performances of calibration and fusion of sub-systems on the development set of CallMyNet corpus.*

		DEVELOPMENT SET			EVALUATION SET		
#	System	EER(%)	act_Cprimary	min_Cprimary	EER(%)	act_Cprimary	min_Cprimary
1	DNN-iVector PLDA 600D (baseline)	17.19	0.8217	0.8097	14.69	1.1588	0.9350
2	DNN-iVector Symmetric SVM 600D	18.26	**0.7599**	**0.7243**	**13.06**	**0.7914**	**0.6894**
3	GMM-iVector PLDA 600D	20.86	0.8438	0.7967	16.09	1.2357	0.9560
4	GMM-iVector PLDA 800D	21.58	0.8297	0.7666	15.86	1.128	0.9362
5	DNN+GMM-iVector DML 1200D	**16.86**	0.7990	0.7607	14.07	0.8245	0.8151
6	DNN-iVector DML 600D	18.78	0.8706	0.8324	15.73	0.8727	0.8714
7	GMM-iVector DML 600D	21.28	0.8466	0.8008	15.86	0.8565	0.8549
8	*fusion of 7 system*	*15.34*	*0.7062*	*0.6688*	*11.32*	*0.6974*	*0.6956*

Table 4: *Performances of final Submission systems under fixed condition in terms of equalized metrics.*

		DEVELOPMENT SET		EVALUATION SET	
Submission	description	act_Cprimary	min_Cprimary	act_Cprimary	min_Cprimary
Submission 1: fixed-primary	fusion of sys1~7 (linear cal.)	0.7062	0.6688	**0.6974**	**0.6956**
Submission 2: fixed-contrastive 1	fusion of sys1~4	0.7177	0.6728	0.7304	0.7278
Submission 3: fixed-contrastive 2	fusion of sys1~7+qm	**0.7029**	**0.6634**	0.7031	0.6972
Submission 4: Open-primary	fusion of sys1~7(PAV cal.)	**0.6116**	**0.5921**	0.8141	0.7067

scenario. Therefore, we explore the complementation of the two front-ends in sub-system 5 and note this strategy is effective.

Finally we note the higher iVector dimension is beneficial (comparing sub-system 4 & 5) at the cost of more computation.

6.3. Subsystem Results

To max out the sub-systems, they are processed with various fusion strategy. Compared with the baseline system (sub-system 1), the fused system (system 8 in Table 3) can relatively boost the min_Cprimary by 25.6%. The fused system is submitted as primary one for fixed condition.

To further optimize system, we explore more variations:

- *Submission 2 (fixed-contrastive 1):* Compared with Submission 1, this one does not include DML. The degraded performance demonstrates the significant contribution from DML.
- *Submission 3 (fixed-contrastive 2):* Compared with Submission 1, we only observe minor improvements on development set. This suggests that more indicative quality measure should be explored.
- *Submission 4 (open-primary):* Although this is for open condition, we use exactly the same data configuration as for fixed condition. Unlike submission 1, we use PAV calibration. Compared with submission 1, we see 11.5% relative gain on development set. At the same time, act_Cprimary is very close to min_Cprimary, which is desirable according to the PAV algorithm [18]. Unfortunately, we do not observe similar trend on the evaluation set. We deduce that this resulted from the fusion tuning data configuration: we use development set as tuning data, which apparently cannot represent the nature of evaluation data. This implies that the PAV-based fusion over-fits the development set.

7. Timing and memory report

The experiment is run on a GPU-CPU Sun Grid Engine (SGE) cluster. The cluster has 10 Tesla M40 GPU cards and five CPUs, each of which has 96 cores. If processed with a single GPU, DNN training takes 192 hours and iVector extractor training takes 240 hours. iVector extraction takes 6mins for each audio file. The scoring takes less than 1min. DNN training takes 20Gb RAM, iVector training takes 100Gb RAM. Given the huge cost of DNN-based system, in most cases, we optimize the sub-components based on GMM-iVector, which can be done within 24 hours. As far as symmetric SVM is concerned, each SVM is a binary classifier. This means we need to train thousands of SVM models for both positive and negative SVMs, which can be speeded up with SGE parallel processing. With the abovementioned hardware configuration, it can be done within 2 hours.

8. Conclusions

Short duration and language mismatch between verification data (include enrollment and test data) and model developing data pose serious challenge to speaker verification. With our proposed methods, we can significantly boost the system performance: Symmetric SVM can better leverage the unlabeled data. Distance metric learning allows us to make full use of the potential of the limited development data. With linear fusion, these two components together with others can relatively improve the min_Cprimary metric by 25.6% against the DNN-iVector PLDA baseline system. As the next step, the utilization of data mining of out-of-domain data will be explored. We also plan to explore the language mismatch in the speaker verification data over Alibaba's ecosystem, in which mandarin is dominating language, but there are still significant amount of people that speak other languages.

9. Acknowledgements

We appreciate the help from I4U consortium under the organizing of Dr. Kong Aik Lee on various aspects from data partition to scoring script sharing, with lots of insightful discussing from the weekly meeting and candid experience sharing among all members. We would like to thank Chunlei Zhang from CRSS for the training data partition suggestion.

10. References

[1] "NIST 2016 speaker recognition evaluation plan," https://www.nist.gov/sites/default/files/documents/2016/10/07/sre16_eval_plan_v1.3.pdf, 2016

[2] D. Snyder, D.G. Romero, and D. Povey. "Time delay deep neural network-based universal background models for speaker recognition." 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU). IEEE, 2015.

[3] H. Sak, A. Senior, and F. Beaufays. "Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition." Computer Science (2014):338-342.

[4] J. Schmidhuber. "A fixed size storage O (n3) time complexity learning algorithm for fully recurrent continually running networks." Neural Computation 4.2, pp. 243-248, 1992

[5] D. Garcia-Romero and C.Y. Espy-Wilson. "Analysis of iVector Length Normalization in Speaker Recognition Systems." In Interspeech, pp. 249-252. 2011.

[6] G. Liu, J.W. Suh, J.H.L. Hansen, "A fast speaker verification with universal background support data selection", Proc. ICASSP2012, Kyoto, Japan, 2012. pp.4793-4796

[7] G. Liu and J.H.L Hansen. "An investigation into back-end advancements for speaker recognition in multi-session and noisy enrollment scenarios." IEEE/ACM Transactions on Audio, Speech, and Language Processing 22, no. 12 (2014): 1978-1992.

[8] V. Hautamaki, etc, "Automatic regularization of cross-entropy cost for speaker recognition fusion", in Proc. INTERSPEECH, Lyon, France, 25-29 Aug.,2013.

[9] R. Saeidi, etc. "I4U submission to NIST SRE 2012: A large-scale collaborative effort for noise-robust speaker verification", in Proc. INTERSPEECH, Lyon, France, 25-29 Aug., 2013.

[10] G. Liu, T. Hasan, H. Boril, J.H.L. Hansen, "An investigation on back-end for speaker recognition in multi-session enrollment", in Proc. ICASSP, Vancouver, Canada, May 25-31, 2013. pp. 7755-7759.

[11] T. Hasan, S. O. Sadjadi, G. Liu, N. Shokouhi, H. Boril, J.H.L. Hansen, "CRSS systems for 2012 NIST speaker recognition evaluation", in Proc. ICASSP, Vancouver, Canada, pp. 6783-6787, 2013.

[12] C. C. Chang and C. J. Lin, "LIBSVM: A library for support vector machines", ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[13] Q. Qian, R. Jin, S. Zhu and Y. Lin, “Fine-grained Visual Categorization via Multi-stage Metric Learning”, in Proc. CVPR, pp. 3716-3724, 2015.

[14] M. Guillaumin, J. Verbeek and C. Schmid, “Is that you? Metric Learning Approaches for Face Identification”, in Proc. ICCV, pp. 498-505, 2009.

[15] X. Fang, N. Dehakand and J. Glass, “Bayesian Distance Metric Learning on iVector for Speaker Verification”, in Proc. INTERSPEECH, pp. 2514-2518, 2013.

[16] W. Ahmad, H. Karnick and R. M. Hegde, “Cosine distance metric learning for speaker verification using large margin nearest neighbor method”, Advances in Multimedia Information Processing, pp. 294–303, 2014.

[17] Q. Qian, R .Jin, J. Yi, L. Zhang and S. Zhu, “Efficient distance metric learning by adaptive sampling and mini-batch stochastic gradient descent (SGD)”, Machine Learning 99 (3), pp. 353-372, 2015.

[18] N. Brümmer, and E. Villiers, 2013, "The Bosaris toolkit: Theory, algorithms and code for surviving the new dcf". arXiv preprint arXiv:1304.2865.

[19] K. Chen, Z. J. Yan, and Q. Huo. "A context-sensitive-chunk BPTT approach to training deep LSTM/BLSTM recurrent neural networks for offline handwriting recognition." International Conference on Document Analysis and Recognition pp.411-415, 2015