Speaker recognition by means of Deep Belief Networks

Vasileios Vasilakakis, Sandro Cumani, Pietro Laface,

Politecnico di Torino, Italy
{first.lastname}@polito.it

1. Abstract

Most state–of–the–art speaker recognition systems are based on Gaussian Mixture Models (GMMs), where a speech segment is represented by a compact representation, referred to as "identity vector" (ivector for short), extracted by means of Factor Analysis. The main advantage of this representation is that the problem of intersession variability is deferred to a second stage, dealing with low-dimensional vectors rather than with the high-dimensional space of the GMM means.

In this paper, we propose to use as a pseudo-ivector extractor a Deep Belief Network (DBN) architecture, trained with the utterances of several speakers. In this approach, the DBN performs a non-linear transformation of the input features, which produces the probability that an output unit is on, given the input features. We model the distribution of the output units, given an utterance, by a reduced set of parameters that embed the speaker characteristics.

Tested on the dataset exploited for training the systems that have been used for the NIST 2012 Speaker Recognition Evaluation, this approach shows promising results.

2. Introduction

Many resources have been devoted in the last few years to Auto Associative Neural Networks (AANNs), Bottleneck Networks, and Deep Belief Networks (DBNs) as possible frameworks for innovative solutions to speech and speaker recognition problems. DBNs have been successfully used for speech recognition [1], rising increasing interest in the DBNs technology [2]. AANNs, trained to reconstruct the input features, or even simple Neural Networks classifiers, have been used to compress in a bottleneck layer the information given by a window including a suitably wide context. The outputs of the bottleneck units are then used as new features for training traditional classifiers such as Hidden Markov Models for speech [3][4], or as features for GMM based speaker recognition systems [5].

AANNs have also been used, without exploiting a wide input context, but still using the compression layer as a feature extractor, for training an ivector based speaker recognition system [6]. In another approach, the weights of the bottleneck layer have been adapted to the speaker of the current utterance and then used as ivectors [7]. Although some of the mentioned techniques use complex architectures and computationally expensive optimization procedures, none of them is able to reach state–of–the–art performance in speaker recognition. On the contrary, the published results lay well behind the ones obtained by the "standard" ivector systems using a Probabilistic Linear Discriminant Analysis (PLDA) classifier [8],[9]. Preliminary experiments with DBNs have been reported in [10],[11] using i–vectors as input features for DBN based classifiers.

In contrast with these approaches, which try to estimate a huge number of parameters with a relatively small number of ivectors, our approach tries to extract a pseudo-ivector directly from the frames of the speech segment, i.e. we leave to a PLDA classifier the task of discriminating among the speakers. In particular, our approach aims at extracting a compact information, which summarize the speaker characteristic given an utterance, directly from the output units of a stack of Restricted Boltzmann Machines (RBMs). In this paper, we will refer to this stack of RBMs as a DBN. The DBN performs a non–linear transformation of the input features, and produces the probability that an output unit is active, given a wide–context of input frames. We model the distribution of each output unit, given an utterance, by a few set of parameters which embed the speaker characteristics because they average the acoustic content of the utterance.

In this work we tested several techniques to model these distributions and to extract pseudo-ivectors. We trained and tested PLDA classifiers using these pseudo-ivectors on the dataset exploited for training the systems that have been used for the NIST 2012 Speaker Recognition Evaluation [12]. Although this approach does not obtain state-of-the-art results, it is an attempt to go beyond the use of the DBN as an ivector classifier, or as a bottleneck feature extractor.

The paper is organized as follows: Section 2 briefly recalls the GMM, the ivector and the PLDA models for speaker recognition. Section 3 introduces the Restricted Boltzmann Machine (RBM) model, and illustrates our approach for pseudo-ivector extraction based on the analysis of the probability distribution of the DBN output units. Section 4 details the computation of different pseudo-ivectors from the DBN output nodes probability distributions. Section 5 is devoted to the illustration of the training and test datasets, and to the experimental results. Finally, in Section 6 we draw our conclusions.

3. GMM Speaker models

Our reference models in this work are the state-of-the-art speaker Gaussian Mixture Models (GMMs), which are used to estimate statistics that allow obtaining a low–dimensional representation of a speech segment, the so–called "identity vector" or ivector [13][14]. An ivector is a compact representation of a GMM supervector [15], representing both the speaker and channel characteristics of a given speech segment, which captures most of the supervector variability. The ivector representation constrains the GMM supervector **s** to live in a single subspace according to:

$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{w}\,, \tag{1}$$

where ***m*** is the Universal Background Model (UBM) supervector, **T** is a low-rank rectangular matrix, of *C*x*F* rows and *M* columns, and *C* and *F* are the number of GMM components and feature dimensions, respectively. The *M* columns of **T** are vectors spanning the subspace including important inter and intra–speaker variability in the supervector space, and **w** is a latent variable of size *M* with standard normal distribution. A Maximum-Likelihood estimate of matrix **T** is obtained by minor modifications of the Joint Factor Analysis approach [16][17].

Ivectors, in conjunction with a PLDA classifier, allow state of the art results to be obtained. A Gaussian PLDA system, implemented according to the framework illustrated in [8], has been used in this work.

The details of the features, and datasets used for training these models are described in Section 6.

4. Restricted Boltzmann Machines

A Boltzmann machine is a generative Neural Network that can learn a probability distribution over its set of binary inputs. A Restricted Boltzmann Machine is a variant of a Boltzmann machine with a hidden layer **h** and a visible layer **v**, and without hidden-to-hidden or visible-to-visible connections, as depicted in Figure 1.

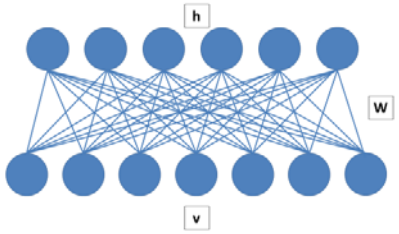

Figure 1. A Restricted Boltzmann Machine

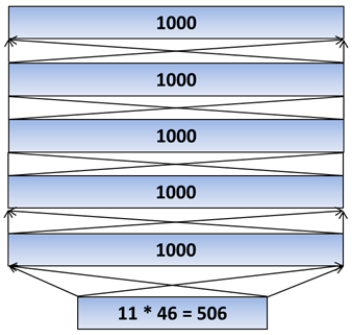

Figure 2: Topology of a 5 layer RBM. The visible layer of the first RBM takes a context of 11 frames consisting of 46 parameters. All the other layers have 1000 units.

Since the RBM has the shape of a bipartite graph, with no intra-layer connections, the hidden unit activations are mutually independent, given the visible unit activations, and vice-versa. RBMs are trained by means of the contrastive divergence technique [18], which is also able to deal with continuous input values, as the ones that have been used in our experiments.

The activation probability of a hidden unit *j* is given by:

$$p(h_j = 1|v) = \sigma\left(b_j + \prod_{i=1}^{n} W_{ij}\, v_i\right), \tag{2}$$

where

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{3}$$

denotes the sigmoid function.

Thus, after the network has been trained, the activation value of a unit of the first RBM hidden layer is its activation probability given the values plugged to its visible units. This single layer of binary features can be used as data for training a second RBM, and this procedure can be performed for the desired number of layers. Each layer of features captures higher-order correlations between the activities of the units of the previous layer.

The main idea in our proposal is to train a DBN consisting of a stacked set of RBMs using as input data a wide context of the frames of several hundred speakers segments, as shown in Figure 2, where the number of trained RBMs is 5, the input layer takes a context of 11 frames consisting of 46 parameters, and each hidden layer has 1000 units. The first layer is a Gaussian-Bernoulli RBM trained on acoustic features, whereas the others layers of the DBN are Bernoulli-Bernoulli RBMs. Since this DBN can be seen as a UBM with a very large number of mixture components, we can try to exploit the probability distributions from the activation probabilities of each hidden unit of the top RBM (we will refer to these units as the "output nodes"). Our assumption is that these probability distributions, and their shape, carry information about the speaker identity, because the phonetic content of the segment, for long enough utterances, is averaged. It is worth noting that the DBN is

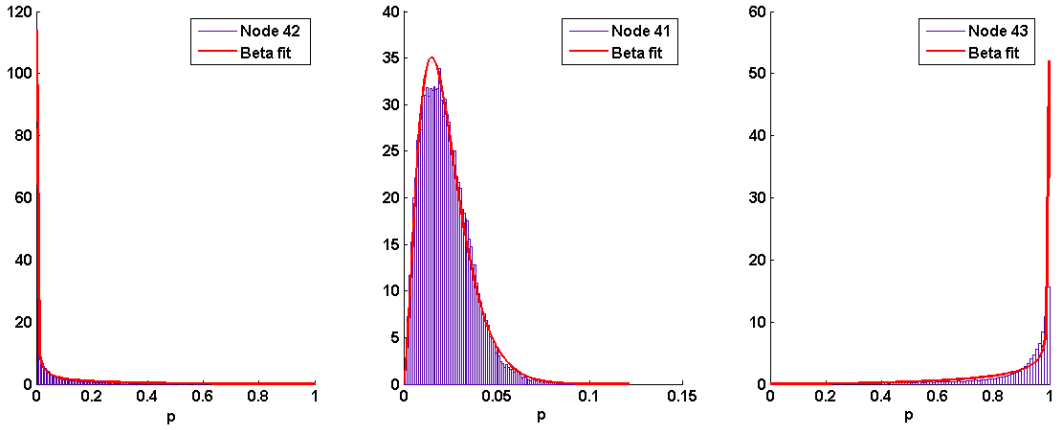

Figure 3: Distribution of the activation probability and the Beta fitting pdf for three output nodes.

not further retrained with a different objective function, leaving to the PLDA classifier the task of discriminating among the speakers.

Since the outputs of the net are highly correlated, Principal Component Analysis is performed to obtain a pseudo–ivector with low dimensions, to be used as observation sample for the standard PLDA model, which takes also care of the intersession variability.

5. Pseudo-ivectors

Three main sets of pseudo-ivectors have been extracted, based on the analysis of the probability distribution of the output units of the same DBN.

5.1. Empirical mean and variance

The simplest pseudo-ivector extraction approach computes the average value of the probability that a given output unit j is active:

$$\mu_j = \frac{1}{T}\sum_t (p_{tj}=1 \mid \mathrm{v}) \,, \tag{4}$$

where T is the number of frames of the speaker segment, without performing any decorrelation of the obtained μ_j, i.e., using as pseudo-ivector a 1000-dimensional vector $\boldsymbol{\mu}$. The results obtained using these pseudo-ivectors were, as expected, not good compared with the pseudo-ivectors obtained by reducing their dimension by means of a PCA projection. Appending to $\boldsymbol{\mu}$ the vector of the variances of each output node probability, and then performing PCA, gives far better results at the cost of increasing the dimension of the pseudo-ivectors.

5.2. Beta distribution fitting

Figure 3 shows the distribution of the probabilities of three output nodes, computed for a file including 28596 voice frames. Similar plots are obtained for the other output nodes, and for other files. These distributions are not at all Gaussian. This is not surprising because the output layer produces activation probabilities, i.e., the probability that the output of node j is active, given input $\boldsymbol{x}_t$. The figure shows that the activation probability of a specific node is concentrated near 0 or 1, and that its distribution has a shape that can be better fit by a Beta distribution. In particular, given an input $\boldsymbol{x}_t$, the j–th output $\boldsymbol{y}_{t,j}$ of the network is active with a probability $p_{t,j}$, which depends on the input value and the network weights.

In the following, we assume that the activation probabilities $p_{t,j}$ are realizations of a random variable P_j, and that each random variable P_j follows a Beta distribution with parameters α_j, β_j

$$P_j \sim B(\alpha_j, \beta_j) \,,$$

and that the set of P_j are independent.

The pair of parameters of the Beta distribution (α_j, β_j) can be estimated by maximizing the log–likelihood of the set of T observations $\{ p_{1,j}, \ldots, p_{Tj} \}$ of a speech segment as:

$$\alpha_j, \beta_j = argmax_{\alpha,\beta} \log L(p_{1j}, p_{2j}, \cdots, p_{Nj} \mid \alpha, \beta) \,. \tag{5}$$

The log–likelihood in (5) is given by:

$$\begin{aligned} \log L(p_{1j}, p_{2j}, \cdots, p_{Nj} \mid \alpha, \beta) = & -T \log B(\alpha, \beta) \\ & + (\alpha - 1) \sum_t \log p_{tj} \\ & + (\beta - 1) \sum_t log(1 - p_{tj}) \,, \end{aligned} \tag{6}$$

which can be rewritten as:

$$\begin{aligned} \log L(p_{1j}, p_{2j}, \cdots, p_{Nj} \mid \alpha, \beta) = & -N \log B(\alpha, \beta) + \\ & (\alpha - 1) L_p + \\ & (\beta - 1) L_m \,, \end{aligned} \tag{7}$$

where L_p and L_m are the sufficient statistics:

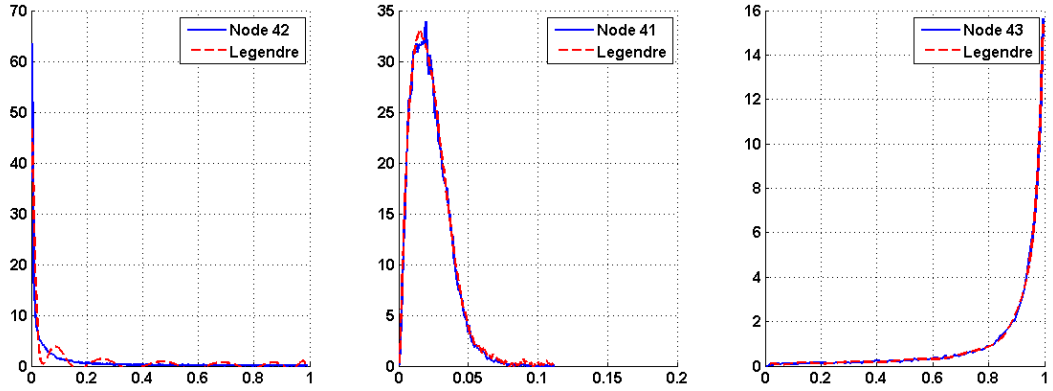

Figure 4: Distribution of the activation probability and Legendre regression for three output nodes.

$$L_p = \sum_t \log p_{tj}$$
$$L_m = \sum_t \log(1 - p_{tj}) \,. \tag{8}$$

These sufficient statistics can be easily obtained from the network outputs. We can find a ML solution for obtaining every pair (α_j, β_j) by maximizing the log–likelihood (7). These parameters can be estimated by means of a generic optimizer. In particular, we used the LBFG algorithm [19], a quasi-Newton optimizer, with a regularization which avoids that the mean of a Beta distribution, computed as:

$$\mu_j = \frac{\alpha_j}{\alpha_j + \beta_j} \,, \tag{9}$$

differs too much with respect to the empirical distribution mean. Then, three different sets of pseudo-vectors have been extracted from these statistics by performing a 400-dimensional PCA projection of three vectors consisting of:

- the concatenation of all the (α_j, β_j) pairs,
- the vector of the mean of the Beta distributions $\boldsymbol{\mu}$,
- the vector of the mean of the log probability of each output unit L_p (8).

5.3. Legendre Polynomial fitting

As shown in Figure 3, for the output nodes 42 and 43, the Beta fitting pdf does not approximate very well actual distributions near the extreme values (0 or 1). Thus, a third pseudo-ivector extractor has been devised, based on Legendre polynomial fitting. In this approach, the distribution of the probability of each output node is approximated by taking the first K terms of a Legendre polynomial expansion estimated to fit the distribution. In our experiments we used Legendre polynomial expansions with K=13 terms. Figure 4 shows the distribution of the activation probability and the Legendre regression for the same output nodes illustrated in Figure 3. Compared to Beta fitting, Legendre regression is more accurate, although it has some difficulty with very sharp distributions, such as the distribution of output node 42. Legendre polynomial fitting requires a substantially larger number of coefficients compared to the first and second order statistics, or to the concatenation of the (α_j, β_j) pairs.

6. Experimental settings and results

The features used in this work for training the models consist of 46 parameters, 19 MFCC coefficients (c1-c19), obtained from the output of a 300-3400Hz Mel Filter-bank, 19 delta (Δc0- Δc18), and 8 double-delta (ΔΔc0-ΔΔc7). These features were computed with a frame rate of 100 observation vectors per second, and were subject to short term gaussianization [20] computed on a 3 sec sliding window applied to speech frames only.

Our reference system uses GMMs consisting of 2048 diagonal Gaussian mixtures. Gender dependent UBMs were trained using the conversations of the NIST SRE 2006, 2008 and 2010. The training set includes 737 hours of speech selected from the 21780 conversations of 1095 female speakers and 512 hours from 15726 conversations of 723 male speakers. Matrix **T** has been obtained using the same dataset. The dimension of the ivectors has been set to 400.

We trained PLDA models with full–rank channel factors, using 200 dimensions for the speaker factors. The ivectors used for the PLDA models are L2 normalized. PLDA training was performed using utterances without any added noise. The training and test datasets for development were selected from the male SRE 2012 training data of the target models, eliminating the 10sec and the summed conversation utterances, and taking care that highly correlated segments (e.g. same interview from different microphones) were all assigned either to the training or to the test set. The development training set finally included

Table 1: Performance of the reference GMM ivector PLDA system, and of three other types of pseudo-ivectors extracted from the statistics of the output nodes of a DBN.

System	% EER	DCF08	DCF10
GMM (reference)	**0.45**	**18**	**87**
1000 means no PCA	2.91	105	237
Dim. 400 PCA projection	**0.81**	**40**	**190**
Dim. 400 PCA projection from the means of the hidden units of the third DBN layer	1.03	47	211

572 hours of speech selected from the 16850 conversations of 1095 female speakers and 391 hours from 12070 conversations of 723 male speakers. The partition of the SRE 2012 training data not used as PLDA training set was used as development test set, for a total of 8204 true and more than 15 million impostor trials.

It is worth noting that in this work we scored every male test segment against all the others, irrespective of the channel or noisy conditions defined in the NIST 2012 SRE [21]. The reported results are obtained without any score normalization

Although the distribution of each output node probability is not Gaussian, the information carried by the distribution mean is the easiest to be computed, thus Table 1 shows the results of experiments aiming at evaluating pseudo-ivectors extracted from the means only, in terms of %EER, minDCF08, and minDCF10 (x 1000). In particular, the first line of the table shows the performance of a GMM system, quite well aligned with state-of-the art systems, which is our reference. The results in the second and third line of the Table 1 make evident the importance of the decorrelation of the mean output probabilities. The performance of the mean based pseudo-ivector system, highlighted in the third row of the table, is the reference for all the other pseudo-ivector systems. The results of the last row support the hypothesis that deeper networks are able to capture more information with respect to shallow networks. Here, the performance using the output of the 5 layer DBN is better than the one obtained from the output of the third layer of the same DBN.

As far as the statistics obtained by estimating the Beta fitting distribution is concerned, the results are summarized in Table 2. Results are reported for pseudo-ivectors obtained projecting to 400 dimensions the concatenation of

Table 2: Performance of 400-dimensional pseudo-ivectors obtained using Beta distribution statistics.

PCA Dim 400	% EER	DCF08	DCF10
$\alpha + \beta$	0.91	44	181
L_p	0.89	45	190
$\boldsymbol{\mu}(\alpha_i, \beta_i)$	0.77	39	170

all pairs (α_j, β_j) estimated by the regularized LBFG algorithm, the average of the log probabilities L_p, and finally the values of the means obtained as a function of the pairs (α_j, β_j).

The first two set of parameters are similar to the ones obtained with the means only, reported in Table 1, whereas an improvement is achieved by using the means computed from the pairs (α_j, β_j) estimated by the regularized LBFG algorithm.

Again, although the distribution of the output probabilities is not Gaussian, rough information about the shape of the each distribution can be given by its variance. Since in this case the number of parameters doubles, the dimension of the PCA projections has been increased. The first column of Table 3 shows the results obtained by using the concatenation of means and variances, and PCA projection with increasing dimensions. The performance keeps improving up to 1200-dimensional pseudo-ivectors.

Since obtaining the pairs (α_j, β_j) is more expensive than just concatenating the means and variances, and the LBFG algorithm requires setting the regularization parameter, the pseudo-ivector based on Beta distributions were no more exploited in the remaining experiments. Table 3 reports in its second column the results for the pseudo-ivectors extracted from the vector concatenating the set of 13 Legendre coefficients fitting each output node distribution. It can be observed that the system based on Legendre polynomial coefficients uses a much larger number of parameters, but gives similar or slightly worse results with respect to system based on the concatenation of means and variances for every dimension of the pseudo-ivectors. However, as shown in the third column of the Table 3, the concatenation of all these parameters improves the performance, mostly the % EER. Compared to the means alone, this combination improves the performance by 41%, 25%, and 19% for the % EER, minDCF08, and minDCF10, respectively, but is still well behind the GMM reference system.

Table 3: Performance of three types of pseudo-ivector extractors.

PCA Dim.	means + variances (1000 + 1000)			Legendre coefficients (13000)			means + variances + Legendre coefficient (15000)		
	% EER	DCF08	DCF10	% EER	DCF08	DCF10	% EER	DCF08	DCF10
800	0.7	38	170	0.75	37	182	0.6	33	162
1000	0.68	37	160	0.72	37	176	**0.58**	**32**	**160**
1200	0.71	36	156	0.75	37	172	0.64	33	154

7. Conclusions

We have proposed to use a stack of Restricted Boltzmann Machines Deep Belief Network as a pseudo-ivector extractor. The distribution of the output units, given an utterance, have been modeled by a reduced set of parameters that embed the speaker characteristics.
The reported results are not comparable to the state-of-the-art, still we consider them promising considering that they have been obtained using a not yet mature technology. Our results are comparatively similar to the ones obtained by the systems so far proposed in the literature because the latter compare their performance only with results obtained by classifiers based on LDA and Cosine Distance scoring, which are known to perform worse than the state-of-the-art PLDA systems.

8. References

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A.Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for acoustic modeling in speech recognition," IEEE Signal Processing Magazine, vol. 28, no. 6, pp. 82–97, 2012.

[2] L. Deng, G. Hinton, B. Kingsbury, "New types of Deep Neural Network Learning for speech recognition and relate applications: An overview", in Proceedings ICASSP 2013, pp. 8599-8603, 2013.

[3] F. Grezl and P. Fousek, "Optimizing bottleneck features for LVCSR," in Proceedings of ICASSP 2008, pp. 4729–4732, 2008.

[4] C. Plahl, R. Schluter, and H. Ney, "Hierarchical bottleneck features for LVCSR," in Proceedings of Interspeech 2010, pp. 1197–2000, 2010.

[5] S. Yaman, J. Pelecanos, and R. Sarikaya, "Bottleneck features for speaker recognition," in Proceedings of Odyssey 2012, pp. 105–108, 2012.

[6] S.Garimella and H. Hermansky, "Factor Analysis of Auto-Associative Neural Networks with application in speaker verification," *IEEE Transactions on Neural Network and Learning Systems*, vol. 24, no. 4, pp. 522–528, 2013.

[7] S. Thomas, H. Mallidi, S. Ganapathy, and H. Hermansky, "Adaptation transforms of Auto-Associative Neural Networks as features for speaker verification," in Proceedings of Odyssey 2012, pp. 98–104, 2012.

[8] N. Brummer and E. de Villiers, "The speaker partitioning problem," in *Proc. Odyssey 2010*, pp. 194–201, 2010.

[9] P. Kenny, "Bayesian speaker verification with Heavy–Tailed priors," in Keynote presentation, Odyssey 2010, Available at http://www.crim.ca/perso/patrick.kenny/kenny_Odyssey2010.pdf.

[10] T. Stafylakis, P. Kenny, M. Senoussaoui, and P. Dumouchel, "Preliminary investigation of Boltzmann Machine classifiers for speaker recognition," in Proceedings of Odyssey 2012, pp. 109–116, 2012.

[11] M. Senoussaoui, N. Dehak, P. Kenny, R. Dehak, and P. Dumouchel, "First attempt at Boltzmann Machines fo speaker recognition," in Proceedings of Odyssey 2012, pp. 1117–121, 2012.

[12] D, Colibro, C. Vair, K. Farrell, N. Krause, G. Karvitsky, S. Cumani, P. Laface, "Nuance - Politecnico di Torino's 2012 NIST Speaker Recognition Evaluation System", in Proc. Interspeech 2013, pp. 1599–1603, 2013.

[13] N. Dehak, R. Dehak, P. Kenny, N. Brummer, and P. Ouellet, "Support Vector Machines versus fast scoring in the low-dimensional total variability space for speaker verification," in Proceedings of Interspeech 2009, pp. 1559–1562, 2009.

[14] 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, 2011.

[15] D. A. Reynolds, "Speaker identification and verification using Gaussian Mixture Speaker Models," *Speech Communications*, vol. 17, no. 1–2, pp. 91–108, August 1995.

[16] P. Kenny, "Joint factor analysis of speaker and session variability : Theory and algorithms". Technical report CRIM-06/08-13, CRIM, 2005

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

[18] G. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, Vol. 14, pp. 1771–1800, 2002.

[19] R. H. Byrd, P. Lu., J. Nocedal, C. Zhu, "A Limited Memory Algorithm for Bound Constrained Optimization". *SIAM J. Scientific Computation*, n.16, vol. 5, pp.1190–1208, 1995.

[20] J. Pelecanos, and S. Sridharan, "Feature Warping for Robust Speaker Verification," in Proc. 2001: A Speaker Odyssey, pp. 213-218, 2001.

[21] National Institute of Standards and Technology, "NIST speech group web," http://www.nist.gov/itl/iad/mig/upload/NIST_SRE12_evalplan-v17-r1.pdf