Available online at www.sciencedirect.com

ScienceDirect

Computer Speech & Language 47 (2017) 16–29

www.elsevier.com/locate/csl

Restricted Boltzmann machines for vector representation of speech in speaker recognition☆

Omid Ghahabi*, Javier Hernando

TALP Research Center, Department of Signal Theory and Communications, Universitat Politecnica de Catalunya, Barcelona 08034, Spain

Received 26 October 2016; received in revised form 2 May 2017; accepted 19 June 2017

Abstract

Over the last few years, i-vectors have been the state-of-the-art technique in speaker recognition. Recent advances in Deep Learning (DL) technology have improved the quality of i-vectors but the DL techniques in use are computationally expensive and need phonetically labeled background data. The aim of this work is to develop an efficient alternative vector representation of speech by keeping the computational cost as low as possible and avoiding phonetic labels, which are not always accessible. The proposed vectors will be based on both Gaussian Mixture Models (GMM) and Restricted Boltzmann Machines (RBM) and will be referred to as GMM−RBM vectors. The role of RBM is to learn the total speaker and session variability among background GMM supervectors. This RBM, which will be referred to as Universal RBM (URBM), will then be used to transform unseen supervectors to the proposed low dimensional vectors. The use of different activation functions for training the URBM and different transformation functions for extracting the proposed vectors are investigated. At the end, a variant of Rectified Linear Units (ReLU) which is referred to as variable ReLU (VReLU) is proposed. Experiments on the core test condition 5 of NIST SRE 2010 show that comparable results with conventional i-vectors are achieved with a clearly lower computational load in the vector extraction process.

Keywords: Restricted Boltzmann machine; Deep learning; Variable rectified linear unit; Speaker recognition; GMM−RBM vector; i-vector

1. Introduction

The low dimensional representation of a speech utterance based on the factor analysis technique is well-known as i-vector (Dehak et al., 2011a). Over the past few years, i-vectors have shown a great performance not only in speaker recognition but also in other applications (e.g., Dehak et al., 2011b; Bahari et al., 2012; Xia and Liu, 2012). Two commonly used scoring techniques for i-vectors are cosine distance (Dehak et al., 2010; 2011a) and Probabilistic Linear Discriminant Analysis (PLDA) (Prince and Elder, 2007; Kenny, 2010). PLDA scoring leads to a superior performance but needs speaker-labeled background data which is costly and not accessible easily.

☆ This paper has been recommended for acceptance by Roger Moore.

* Corresponding author.
E-mail address: omid.ghahabi@upc.edu (O. Ghahabi).

http://dx.doi.org/10.1016/j.csl.2017.06.007

Motivated by the success use of Deep Learning (DL) in other speech processing applications (e.g., Mohamed et al., 2010; Dahl et al., 2012; Mohamed et al., 2012; Hinton et al., 2012; Senior et al., 2015), DL techniques have also been used in speaker recognition for different purposes. For example, DL techniques have been applied as a backend on i-vectors (Stafylakis et al., 2012b; Senoussaoui et al., 2012; Stafylakis et al., 2012a; Novoselov et al., 2014; Ghahabi and Hernando, 2014a; 2014b; 2017), used in the i-vector extraction algorithm (Lei et al., 2014; Kenny et al., 2014; Mclaren et al., 2015; Richardson et al., 2015; Liu et al., 2015; Campbell, 2014; Garcia-Romero et al., 2014), and also employed for compact representation of speech signals (Vasilakakis et al., 2013; Variani et al., 2014; Liu et al., 2015; Ghahabi and Hernando, 2015; Safari et al., 2016) and discriminative feature classification (Safari et al., 2015).

DL technology has been used in the i-vector extraction algorithm in two ways. First, a Deep Neural Network (DNN) has been used for acoustic modeling rather than the typical Gaussian Mixture Model (GMM) (Lei et al., 2014; Kenny et al., 2014; Campbell, 2014; Richardson et al., 2015; Garcia-Romero et al., 2014; Liu et al., 2015). Second, conventional spectral features have been replaced or appended by the so-called DNN bottleneck features and then a DNN or a GMM has been used as an acoustic model (Mclaren et al., 2015; Richardson et al., 2015; Liu et al., 2015). It has been shown that the best results are obtained when spectral features are appended by bottleneck features and a GMM is used as an acoustic model (Mclaren et al., 2015; Richardson et al., 2015; Lozano-Diez et al., 2016). However, the main problem is that the use of DNN as either an acoustic model or bottleneck feature extractor increases highly the computational cost of the i-vector extraction process. Moreover, in both cases phonetic labels are required for DNN training, which are not always accessible.

On the other hand, only a few works have tried to make use of DL techniques to build a compact representation of speech signals without using the conventional i-vector algorithm. In Vasilakakis et al. (2013), Variani et al. (2014), Liu et al. (2015), a deep architecture is trained using background feature vectors. Then the feature vectors of a given utterance are forward-propagated and the mean of the posterior probabilities of a particular hidden layer (Variani et al., 2014) or a PCA dimension reduced version of them (Liu et al., 2015), or a PCA dimension reduced version of the mean vectors (Vasilakakis et al., 2013) are considered as a new compact representation. In Safari et al. (2016), the parameters of the adapted networks are stacked to build a supervector. Then the dimension of the new supervectors are reduced by PCA. In Ghahabi and Hernando (2015), the authors used the GMM supervectors, rather than the feature vectors, as the inputs to a Restricted Boltzmann Machine (RBM). RBM has been used as a dimension reduction stage in that scenario. Although Liu et al. (2015) and Variani et al. (2014) have shown some success in text-dependent speaker recognition, still no significant improvement is reported for text-independent tasks. Moreover, working with DL techniques in feature vector domain is costly.

The aim of this work is to develop an efficient framework for vector representation of speech by keeping the computational cost as low as possible and avoiding phonetic labels. In order to achieve this goal, a global RBM referred to as Universal RBM (URBM) is trained given background GMM supervectors. The URBM tries to learn the total session and speaker variability among background supervectors. It will then be used to transform unseen supervectors to lower dimensional vectors which will be referred to as GMM−RBM vectors.

Compared to the preliminary work presented in Ghahabi and Hernando (2015), whitening in the supervector domain, which is computationally costly, is replaced by warping in the feature vector domain. This change makes possible to obtain higher speaker recognition accuracy, specially in lower dimensional vectors. Moreover, the effect of the type of the activation function for training the URBM and the type of the transformation function for GMM−RBM vector extraction are investigated. At the end, a variation of Linear Rectified Units (ReLU), which will be referred to as variable ReLU (VReLU), is proposed for training the URBM, and then a linear function is used for transformation in the vector extraction stage.

The core condition of NIST SRE 2006 (NIST, 2006) is used for the development and the core condition 5 of NIST SRE 2010 (NIST, 2010) with much bigger background data is used for the test and evaluation. The experiments on the evaluation set shows that the proposed GMM−RBM vectors achieve comparable performance with conventional i-vectors while lower computational cost is required for vector extraction. The conclusion is valid with both cosine and PLDA scoring. Moreover, the combination of GMM−RBM vectors and i-vectors at the score level improves the performance more.

The rest of the paper is organized as follows. Section 2 gives a brief background overview about conventional i-vectors and PLDA. Section 3 describes the proposed GMM−RBM vectors. Section 4 investigate the effect of

activation and transformation functions used, respectively, for URBM training and GMM–RBM vector extraction and discusses the database, baseline systems, and the experimental results. Section 5 concludes the paper.

2. Conventional i-vectors

An i-vector (Dehak et al., 2011a) is a low rank vector representation of a speech utterance. Feature vectors of a speech signal can be represented by a Gaussian Mixture Model (GMM) adapted from a Universal Background Model (UBM). The mean vectors of the adapted GMM are stacked to build the supervector $\boldsymbol{s}$. The supervector can be further modeled as follows,

$$\boldsymbol{s} = \boldsymbol{s}_{ubm} + \boldsymbol{T}\boldsymbol{v} \tag{1}$$

where $\boldsymbol{s}_{ubm}$ is the speaker- and session-independent mean supervector, typically from UBM, $\boldsymbol{T}$ is the total variability matrix, and v is a vector of latent variables. The posterior distribution of v is conditioned on the Baum–Welch statistics of the given speech utterance. The mean of this posterior distribution is referred to as i-vector ω and computed as follows,

$$\boldsymbol{\omega} = \left(\boldsymbol{I} + \boldsymbol{T}^t\Sigma^{-1}\mathcal{N}(\boldsymbol{u})\boldsymbol{T}\right)^{-1}\boldsymbol{T}^t\Sigma^{-1}\widetilde{\mathcal{F}}(\boldsymbol{u}) \tag{2}$$

where $\mathcal{N}(\boldsymbol{u})$ is a diagonal matrix containing the zeroth order Baum–Welch statistics, $\widetilde{\mathcal{F}}(\boldsymbol{u})$ is a supervector of the centralized first order statistics, $\boldsymbol{\Sigma}$ is a diagonal covariance matrix initialized by $\boldsymbol{\Sigma}_{ubm}$ and updated during the factor analysis training, and t denotes the transpose operation. The $\boldsymbol{T}$ matrix is trained using the Expectation–Maximization (EM) algorithm given the Baum–Welch statistics from background speech utterances. More details can be found in Dehak et al. (2011a).

Two main scoring techniques for i-vectors are cosine (Dehak et al., 2010; 2011a) and Probabilistic Linear Discriminant Analysis (PLDA) (Prince and Elder, 2007). PLDA is a more effective technique which performs scoring along with session variability compensation. It assumes that each i-vector can be decomposed as,

$$\boldsymbol{\omega} = \boldsymbol{m} + \Phi\boldsymbol{\zeta} + \varepsilon \tag{3}$$

where $\boldsymbol{m}$ is a global offset, the columns of $\boldsymbol{\Phi}$ are eigenvoices, $\boldsymbol{\zeta}$ is a latent vector having a standard normal prior, and the residual vector ε is normally distributed with zero mean and a full covariance matrix. The model parameters are estimated from a large collection of speaker-labeled background data using an EM algorithm as in Prince and Elder (2007). Within and between class i-vector covariance matrices which are depending only on the model parameters are stored and used for scoring.

3. Proposed GMM–RBM vectors

Recently, the advances in Deep Learning (DL) have improved the quality of i-vectors, but the DL techniques in use are computationally expensive and need phonetic labels for the background data. We propose in this work an alternative vector-based representation for speakers in a less computationally expensive manner with no use of any phonetic or speaker labels.

RBMs are good potentials for this purpose because they have good representational powers and they are unsupervised and computationally low cost. In fact, RBMs are generative networks with two fully connected layers of visible and hidden stochastic units. In this work, it is assumed that the inputs or visible units are GMM supervectors and the outputs or hidden units are the low dimensional vectors we are looking for. The RBM is trained given the background GMM supervectors and will be referred to as a Universal RBM (URBM). The role of the URBM is to learn the total session and speaker variability among the background supervectors. Different types of units and activation functions can be used for training the URBM which will be mentioned in Section 3.2 and evaluated in Section 4 for this application. After training the URBM, the visible–hidden connection weight matrix is used to transform unseen GMM supervectors to lower dimensional vectors which will be referred to as GMM–RBM vectors in this work.

Fig. 1 shows the block-diagram of the proposed framework. The whole process has been divided in three main stages detailed in the following sections, which correspond to each block of Fig. 1. First, GMM supervectors are built from the warped spectral features given the UBM, and then are normalized using the UBM parameters. Second, the

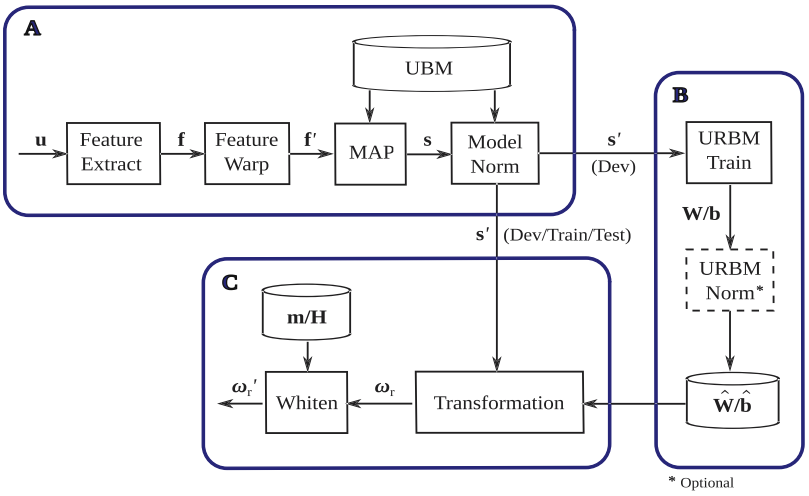

Fig. 1. Block-diagram of the proposed GMM–RBM vector framework. $\boldsymbol{W}$ and $\boldsymbol{b}$ are the parameters of the Universal RBM (URBM), $\boldsymbol{m}$ is the global mean, and $\boldsymbol{H}$ is the whitening matrix obtained on the background GMM–RBM vectors.

background GMM supervectors are used to train the URBM and then it can optionally be normalized to provide an appropriate transformation matrix from the supervectors to the proposed low dimensional vectors. Third, given the unseen GMM supervectors and the parameters of the URBM, GMM–RBM vectors are extracted.

3.1. Feature warping and GMM supervectors

As it is shown in Block A of Fig. 1, input speech signals are first characterized by spectral feature vectors. Afterwards, feature warping is applied to map the distribution of each individual feature to a Gaussian distribution over a time interval based on the Cumulative Distribution Function (CDF). The method assumes that the components of the feature vector are independent and are processed individually as a separate stream. CDF matching is performed over a sliding window of size N and only the central frame of the window is warped. The features in a given window are sorted in ascending order. If the given component value x in the central frame has the rank r ($1 \leq r \leq N$), the warped value $\widehat{x}$ should satisfy (Pelecanos and Sridharan, 2001; Xiang et al., 2002),

$$(r - 1/2)/N = \int_{-\infty}^{\widehat{x}} f(z)dz \tag{4}$$

where the left side is the approximated CDF value of x, the right side is the CDF value of $\widehat{x}$, and $f(z)$ is the Probability Density Function (PDF) of a standard normal distribution (Pelecanos and Sridharan, 2001; Xiang et al., 2002).

It is shown that feature warping helps to compensate undesired session variability in speech signals (Pelecanos and Sridharan, 2001) as well as Gaussianizes the feature distributions. It will be shown in the experimental result section that feature warping has a high impact on the performance of the proposed GMM–RBM vectors.

Warped features are then modeled by a GMM adapted from the background model (UBM). The mean vectors of each adapted GMM are stacked to build a supervector. In order to increase the discrimination power, supervectors are model-normalized using the mean supervector and diagonal covariance matrix of the UBM ($\boldsymbol{s}_{ubm}$ and $\boldsymbol{\Sigma}_{ubm}$),

$$\boldsymbol{s}' = \Sigma_{ubm}^{-1/2}(\boldsymbol{s} - \boldsymbol{s}_{ubm}). \tag{5}$$

Model normalization helps also having zero mean and unit variance for supervectors which is a prior assumption for the training of an RBM with real-valued inputs as it will be described in the next section.

3.2. Universal RBM training

Normalized supervectors obtained on the background data are used to train the URBM (Block B of Fig. 1). The role of the URBM is to learn all session and speaker variability among background supervectors. The URBM

parameters will then be used to transform unseen supervectors to lower dimensional GMM–RBM vectors. Different visible and hidden units, and activation functions can be used for training an RBM (Hinton, 2012). Since the inputs in this application are real-valued supervectors, the visible units will be Gaussian. However, sigmoid and Rectified Linear Units (ReLU) can be used in the hidden layer during the training of the URBM. As it is mentioned in Hinton (2012) and proved by our experiments, training an RBM with both linear hidden and visible units is highly unstable. Therefore, pure linear hidden units are discarded in this work. Given the URBM parameters, any reasonable transformation function could be used to transform unseen supervectors. In this section, the problem of the use of the traditional sigmoid function for both activation and transformation is first addressed and a potential solution is proposed. Then a variant of ReLU, which will be referred to as Variable ReLU (VReLU), is proposed for this application. It will be shown in Section 4 that the proposed VReLU does not suffer from the problems of sigmoid and ReLU.

Fig. 2 shows the histograms of the posterior probabilities of the first hidden unit of the URBM before and after nonlinear transformations. The URBM is trained with traditional sigmoid activation function. The typical sigmoid function and the log sigmoid function, which was used in Ghahabi and Hernando (2015), are employed for the transformation. Other hidden units show also similar behaviors. As it can be seen in this figure, the posterior probability distribution of hidden units after *sigmoid* transformation will be compressed around zero and far from a Gaussian distribution which is ideal for the proposed GMM–RBM vectors. This fact degrades the performance significantly.

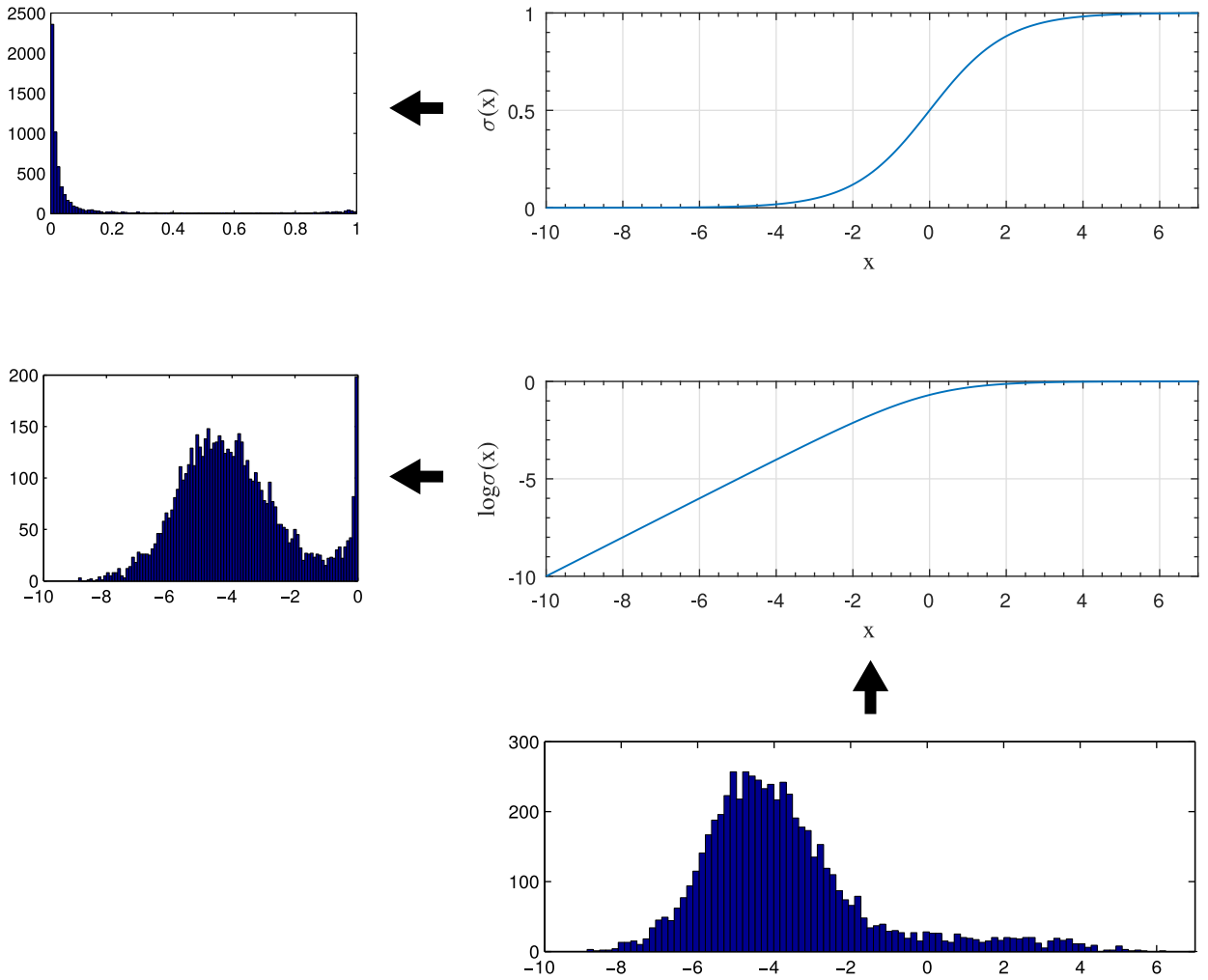

Fig. 2. Histograms of the first hidden unit values before (bottom) and after (left) transformation with *sigmoid* and log *sigmoid* functions. URBM is trained with *sigmoid* hidden units.

The behavior is better in case of log sigmoid function since the most part of the distribution is transformed with linear part of the function, but still there is the same problem for values around zero after transformation. Although the whitening transformation on posterior probabilities afterwards corrects the distributions to some extent, still the performance will be low specifically for sigmoid transformation.

A potential solution can be changing the mean and variance of the posterior probability distributions, before transformation, somehow they fall in the active nonlinear parts of the transformation functions. This can be easily performed through URBM parameter normalization which we have proposed as follows,

$$\widehat{\boldsymbol{W}} = \alpha \frac{\boldsymbol{W}}{\max_{i,j} |w_{ij}|} \tag{6}$$

$$\widehat{b}_i = \beta + \left(b_i - \overline{b}\right) \tag{7}$$

where $\boldsymbol{W}$ is the visible−hidden connection weights, $\boldsymbol{b}$ is the vector of hidden bias terms, α and β are two parameters to control, respectively, the variance and the mean of the posterior probability distributions of hidden units before nonlinear transformation, w_{ij} is the (i, j) element of $\boldsymbol{W}$, and b_i and $\overline{b}$ are the ith element and the mean value of $\boldsymbol{b}$, respectively.

Fig. 3 shows how changing α and β can move the distribution of the posterior probabilities of hidden units to a desired interval. We will show in Section 4 that this movement will improve the quality of the GMM−RBM vectors when the URBM is trained with sigmoid hidden units.

Another alternative unit is ReLU. ReLU is a kind of linear unit for which the negative values are zeroed out. If the URBM is trained with ReLU and the inputs are transformed with linear function after training, none of the above problems will occur. However, as we will show in Section 4, the problem will be that the distribution of posterior probabilities of hidden units will be asymmetric around the mean value, which is not appropriate for PLDA scoring. Therefore, we have proposed in this work a variant of ReLU, which is referred to as variable ReLU (VReLU). In VReLU, the unit values less than the threshold τ are zeroed out, rather than the fixed threshold zero in ReLU. Threshold τ is randomly selected from a normal distribution $N(0, 1)$ for each hidden unit and for each input sample in each training iteration. In fact, VReLU is defined as follows,

$$f(x) = \begin{cases} x & x > \tau \\ 0 & x \leq \tau \end{cases} \quad , \quad \tau \in N(0,1) \tag{8}$$

Fig. 4 compares ReLU and VReLU with both positive and negative values of τ. It will be shown in Section 4 that VReLU solves the asymmetric problem of the posterior probability distributions to a great extent and, therefore, it works better than ReLU when PLDA scoring is used.

The full training algorithm for RBM with sigmoid hidden units can be found in Ghahabi and Hernando (2015). In the following, we only explain the RBM training algorithm with the proposed VReLU. Fig. 5 shows the training steps. The connection weights $\boldsymbol{W}$ are first randomly initialized from $N(0, 0.01)$ and the visible and hidden bias terms ($\boldsymbol{a}$ and $\boldsymbol{b}$, respectively) are set to zero. Given the normalized supervectors $\boldsymbol{s}'$, the posterior probability of the lower

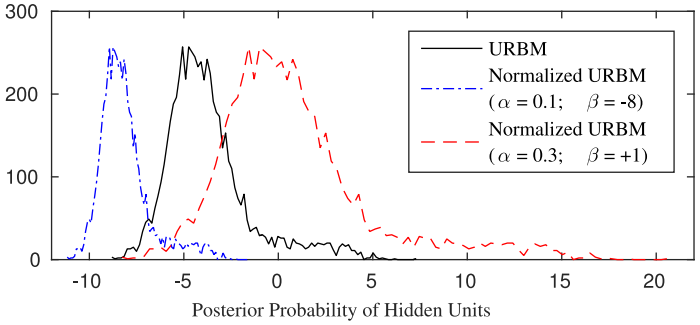

Fig. 3. The histograms of the posterior probabilities of the first hidden unit of URBM and normalized URBM (with two different pairs of α and β) before nonlinear transformation. The histograms are obtained on the background dataset used for development.

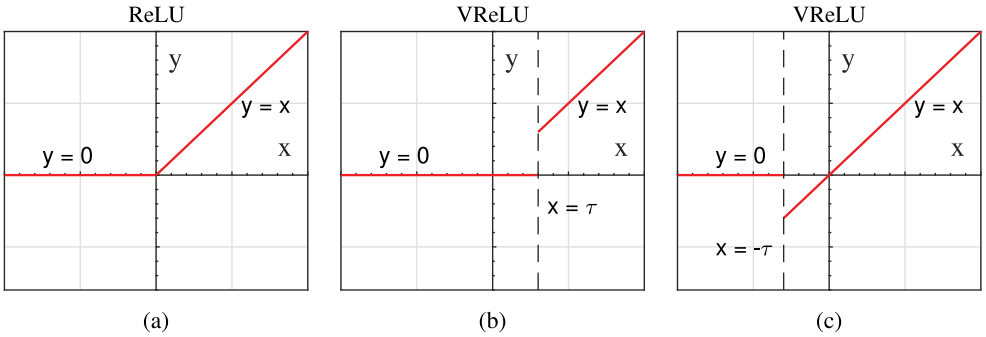

Fig. 4. Comparison of ReLU and proposed VReLU. In each epoch, per each hidden unit and per each input sample, τ is randomly selected from a normal distribution with zero mean and unit variance. (b) and (c) show the two examples of VReLU when τ is positive and negative, respectively.

dimensional hidden vector $\boldsymbol{h}$ is calculated using Eq. (8). Afterwards, supervectors are reconstructed given the hidden unit values. Then the reconstructed supervectors $\boldsymbol{s}'_r$ are used to recalculate the posterior probabilities of hidden units. These three steps, marked in Fig. 5, provide enough information to update the parameters of the network. Actually, the training process is based on a maximum likelihood criterion using the stochastic gradient descent algorithm (Hinton and Salakhutdinov, 2006; Hinton et al., 2006), in which the gradient is estimated by an approximated version of the Contrastive Divergence (CD) algorithm called CD_1 (Hinton et al., 2006; Hinton, 2012).

The training process is summarized as follows,

- Initialize Network Parameters ($\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{a}$)
- CD_1 Steps

1. $\boldsymbol{h} = f(\boldsymbol{b} + \boldsymbol{W}\boldsymbol{s}')$ (9)
2. $\boldsymbol{s}'_r = \boldsymbol{a} + \boldsymbol{W}^t\boldsymbol{h}$ (10)
3. $\boldsymbol{h}_r = f\left(\boldsymbol{b} + \boldsymbol{W}\boldsymbol{s}'_r\right)$ (11)

- Update Network Parameters

1. $\Delta\boldsymbol{W} = \eta \times \left(\boldsymbol{s}'\boldsymbol{h}^t - \boldsymbol{s}'_r\boldsymbol{h}^t_r\right)^t$ (12)
2. $\Delta\boldsymbol{a} = \eta \times \left(\boldsymbol{s}' - \boldsymbol{s}'_r\right)$ (13)
3. $\Delta\boldsymbol{b} = \eta \times (\boldsymbol{h} - \boldsymbol{h}_r)$ (14)

where η is the learning rate and $f(.)$ is the VReLU function calculated as in Eq. (8).

Additionally, a momentum factor is used to smooth out the updates, and the weight decay regularization is used to penalize large weights. The parameters are updated after processing each minibatch and the updating procedure is repeated when all the minibatches are processed.

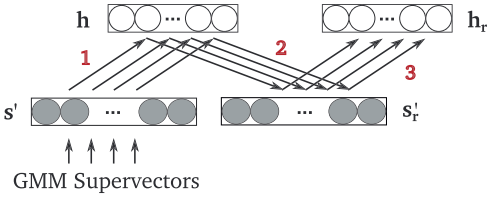

Fig. 5. Training of the Universal RBM (URBM) given background GMM supervectors.

3.3. GMM–RBM vector extraction

Given the GMM supervectors from Block A and the URBM parameters from Block B of Fig. 1, the GMM–RBM vectors are extracted in Block C as follows,

$$\boldsymbol{\omega}_r = \boldsymbol{W}\boldsymbol{\Sigma}_{ubm}^{-1/2}(\boldsymbol{s} - \boldsymbol{s}_{ubm}) \tag{15}$$

As a linear transformation function is used, the hidden unit bias terms $\boldsymbol{b}$ can be easily discarded and only the visible–hidden connection weights $\boldsymbol{W}$ are used for transformation. If we reformulate Eq. (15) based on zeroth and first order Baum–Welch statistics, we will have,

$$\boldsymbol{\omega}_r = \boldsymbol{W}\boldsymbol{\Sigma}_{ubm}^{-1/2}\mathcal{N}^{-1}(\boldsymbol{u})\widetilde{\mathcal{F}}(\boldsymbol{u}) \tag{16}$$

where the relevance factor in map adaptation can also be added to $\mathcal{N}(\boldsymbol{u})$.

Like in case of i-vectors, resulting GMM–RBM vectors are mean normalized and whitened using a mean vector and a whitening matrix obtained on the background data,

$$\boldsymbol{H} = \boldsymbol{V}(\boldsymbol{D} + \epsilon)^{-1/2}\boldsymbol{V}^t \tag{17}$$

where $\boldsymbol{H}$ is the whitening matrix, $\boldsymbol{V}$ is the matrix of eigenvectors, $\boldsymbol{D}$ is the diagonal matrix of the corresponding eigenvalues, and ϵ is a small constant regularization factor added to avoid large values in practice.

3.4. Computational load compared to i-vector

The comparison of Eqs. (2) and (16) implies clearly that GMM–RBM vector extraction needs much less computational load. We compare the computational load in terms of the number of product operations required for extracting an i-vector and a GMM–RBM vector with the same size based on Eqs. (2) and (16). Considering the computational cost for multiplication of two matrices $n \times m$ and $m \times k$ of order $\mathcal{O}(nmk)$ and for a matrix inversion of size $n \times n$ of order $\mathcal{O}(n^3)$, and this fact that $\mathcal{N}(u)$ is diagonal and $\boldsymbol{W}\boldsymbol{\Sigma}_{ubm}^{-1/2}$ in Eq. (16) or $\boldsymbol{T}^t\boldsymbol{\Sigma}^{-1}$ in Eq. (2) are computed offline, the minimum computational load of i-vector and GMM–RBM vector extraction will be $\mathcal{O}(n^3 + (2n^2 + 2n)m)$ and $\mathcal{O}((n+1)m)$, respectively, in which n is the dimension of i-vector/GMM–RBM vector and m is the size of supervector.

Fig. 6 compares the minimum computational load for extracting an i-vector and a GMM–RBM vector for different values of n and m. The figure implies that the number of operations required for extracting a GMM–RBM vector is about $10^{-6} - 10^{-8}$ compared to an i-vector which requires about $10^{-8} - 10^{-11}$ operations. The computational load is of higher importance for online applications in which the frequency of vector extraction is high.

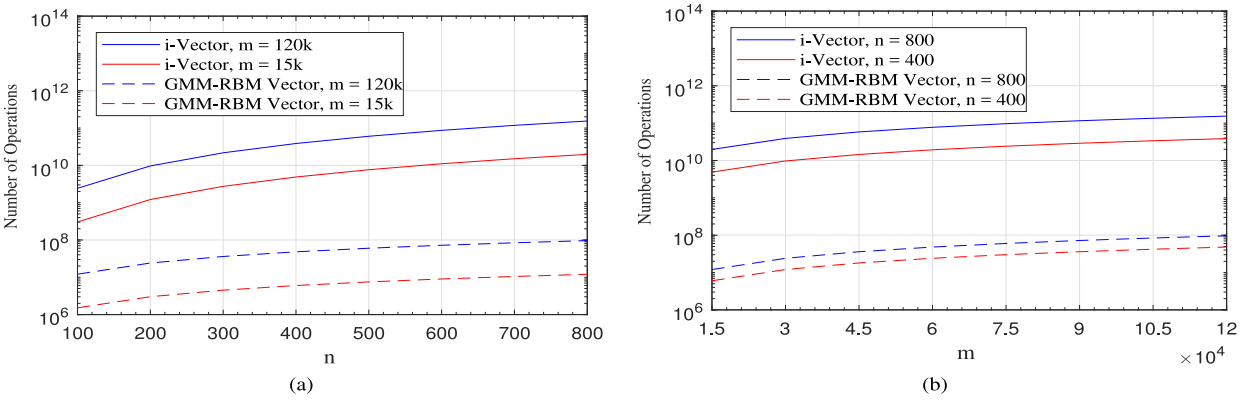

Fig. 6. Comparison of the number of product operations required for extracting an i-vector and a GMM–RBM vector in terms of (a) the size of i-vector/GMM–RBM vector n and (b) the size of supervector m.

4. Experimental results

The details of the database, the setup of the baseline and the proposed approaches, and the experimental results are given in this section. Baseline systems will be based on conventional i-vectors which are scored using either cosine or PLDA techniques. Proposed GMM–RBM vectors are build according to the block-diagrams of Fig. 1. The effect of feature warping, URBM normalization, the type of the activation and transformation functions, as well as the score combination for both cosine and PLDA techniques are shown in this section.

4.1. Setup, baseline, and database

Two sets of database are used for the experiments. For development, the core test condition of the NIST 2006 SRE evaluation (NIST, 2006) is used. It includes 816 target models and 51,068 trials. In both the training and testing phases, the duration of speech in signals is approximately two minutes. The background data includes 6063 speech files collected from NIST 2004 and 2005 SRE corpora. The same background data is used to train UBM, URBM, PLDA, $\boldsymbol{T}$ and whitening matrices.

For evaluation, the NIST 2010 SRE (NIST, 2010), core test-common condition 5, which includes different number of trials involving normal vocal effort conversational telephone speech in training and test, is used. The background data is collected form NIST SRE 2004–2008 and includes 37,600 speech utterances from which 18,140 signals are labeled which are used for PLDA training.

Frequency Filtering (FF) features (Nadeu et al., 2001) are used in the experiments. Like Mel-Frequency Cepstral Coefficients (MFCC), FFs are decorrelated version of log Filter Bank Energies (FBE) (Nadeu et al., 2001). It has been shown that FF features achieve a performance equal to or better than MFCCs (Nadeu et al., 2001). Features are extracted every 10 ms using a 30 ms Hamming window. The number of static FF features is 16 and along with delta FF and delta log energy, 33-dimensional feature vectors are built. Before feature extraction, speech signals are subject to an energy-based silence removal process. After feature extraction, a 3-second sliding window is used for feature warping.

ALIZE open source software (Larcher et al., 2013) is used to build the i-vector baseline systems in which cosine and PLDA scoring techniques are employed. The dimension of i-vectors is 400 and PLDA size for development data is 250 and for evaluation 400. A gender-independent UBM is represented as a diagonal-covariance 512-component GMM.

In the proposed GMM–RBM vector framework, GMMs are adapted from the UBM by a relevance factor of 16. Only mean vectors are adapted. The dimension of supervectors is, therefore, $512 \times 33 = 16{,}896$. Two URBMs with the hidden layer sizes of 400 and 8000 are trained to create GMM–RBM vectors. The bigger one is trained only with sigmoid activation function and is used just for comparing the results with those reported in Ghahabi and Hernando (2015). URBMs with hidden layer size of 400 are trained with sigmoid, ReLU, and the proposed VReLU. The learning rate, the number of epochs, the minibatch size, the weight decay, and the momentum for the URBM, trained with VReLU, are set to 0.0014, 40, 50, 2×10^{-3}, and 0.9, respectively.

Performance is evaluated using the Equal Error Rate (EER), and the minimum of the Decision Cost Function (minDCF) calculated using $C_M = 10, C_{FA} = 1, P_T = 0.01$ for the development experiments (NIST, 2006) and $C_M = 1, C_{FA} = 1, P_T = 0.001$ for the evaluation experiments (NIST, 2010).

4.2. Results

As it was mentioned in Section 1, one of the main differences between this work and our prior work in Ghahabi and Hernando (2015) is discarding whitening in the supervector level and making use of feature warping instead. The results reported in Table 1 imply that when no feature warping is used, whitening in the supervector level helps. This is exactly what we did in Ghahabi and Hernando (2015). However, if feature vectors are warped, the whitening of supervectors is not effective anymore. Moreover, it is time and memory consuming. The best results are obtained when only feature warping is used.

Fig. 7 shows the histograms of the first component of the GMM–RBM vectors obtained with a URBM, which is trained with sigmoid activation function. However, sigmoid, log sigmoid, and linear transformation functions are used for vector extraction. The histograms of other components show similar behaviors. For sigmoid and log sigmoid

Table 1
The effect of feature warping and whitening of input GMM supervectors in the proposed GMM–RBM framework. The numbers in the parentheses indicate the dimensions of GMM–RBM vectors. Results are obtained on the **development** database with cosine scoring.

Input to RBM	Output of RBM	Raw features		Warped features	
		EER	minDCF	EER	minDCF
Whitened supervectors	GMM–RBM vector (8000)	7.58	0.0346	6.90	0.0331
Raw supervectors	GMM–RBM vector (8000)	7.92	0.0379	6.89	0.0323
Raw supervectors	GMM–RBM vector (400)	10.45	0.0475	8.08	0.0383

transformations, the histograms are presented for both URBM and normalized URBM parameters. The normalization parameters α and β in Eqs. (6) and (7) are set to 0.05 and -0.5, respectively. This is to move approximately the posterior distributions into the interval -2 and 2 (Fig. 3) corresponding to the active nonlinear parts of the sigmoid and log sigmoid transformation functions. For both sigmoid and log sigmoid, URBM normalization helps having more Gaussian-like histograms. As it will be shown later, this will increase the performance of GMM–RBM vectors when URBM is trained with sigmoid hidden units.

Fig. 8 shows the same histograms for GMM–RBM vectors for which URBM is trained with ReLU and VReLU and linear transformation is used in both cases. The figure implies that the histograms are asymmetric in case of

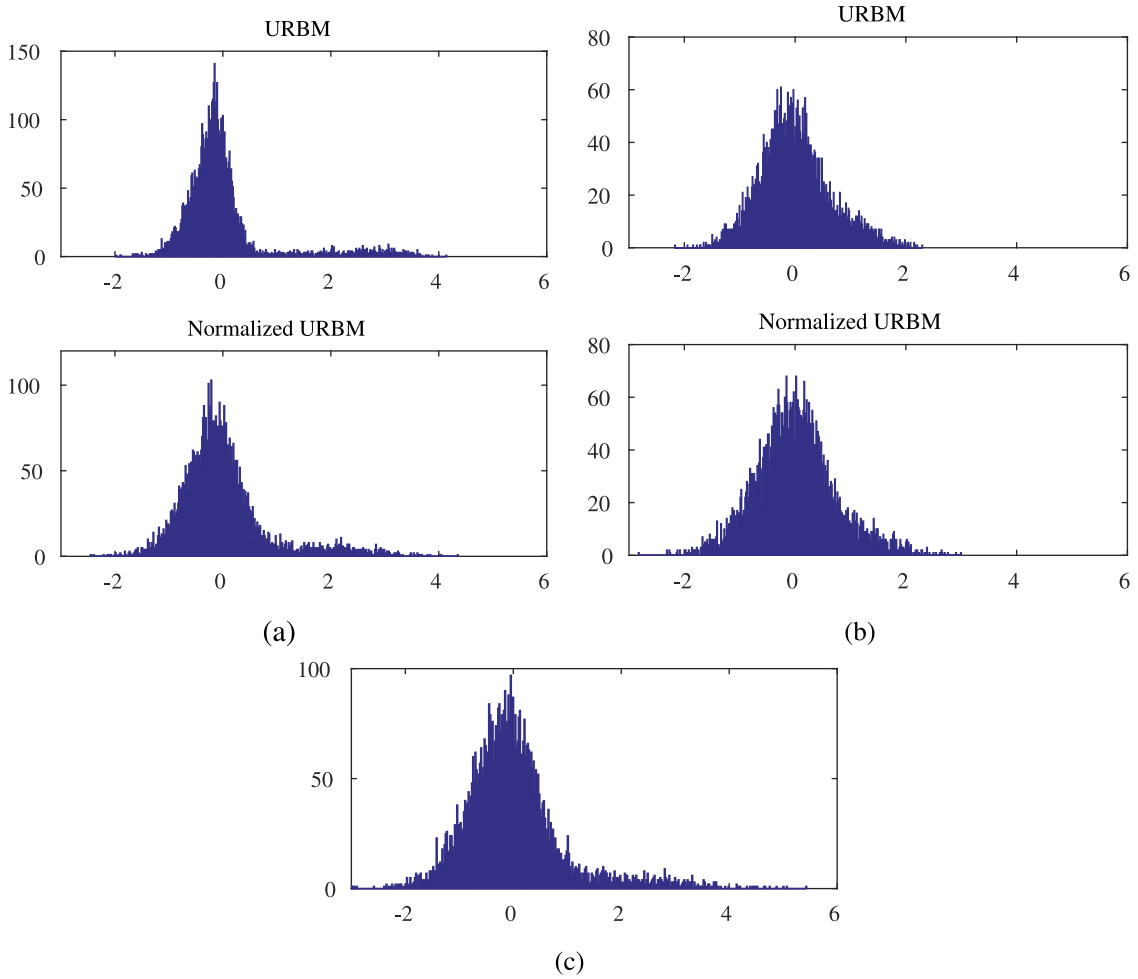

Fig. 7. Comparison of the histograms of the first component of the background GMM–RBM vectors obtained with *sigmoid* activation function and transformation functions of (a) *sigmoid*, (b) log *sigmoid*, and (c) linear.

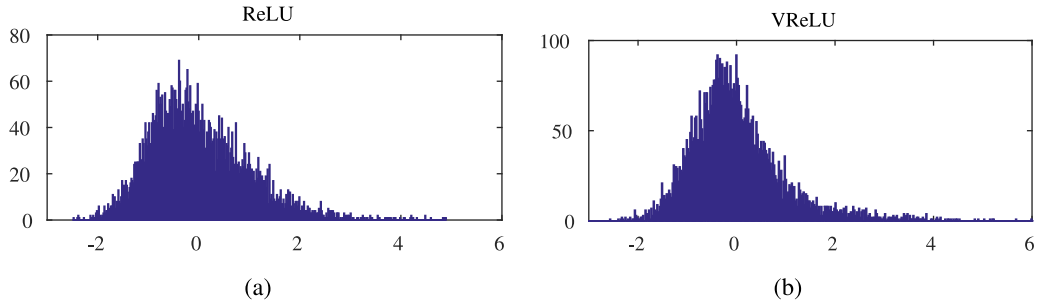

Fig. 8. Comparison of the histograms of the first component of the background GMM–RBM vectors obtained with (a) ReLU and (b) the proposed VReLU activation functions and linear transformation function.

ReLU which is due to the training process in which the hidden units are encouraged having positive values. On the other hand, the random threshold τ proposed in VReLU makes it possible having both positive and negative hidden values during the training process. This improves the histograms as shown in Fig. 8.

Table 2 compares the performance of GMM–RBM vectors extracted by different URBMs and transformation functions. The comparison is based on both cosine and PLDA scoring. As it was expected, the worse results are for sigmoid hidden units and transformation function. The URBM normalization improves significantly the performance of these vectors. The use of log sigmoid itself performs better than sigmoid as discussed for Figs. 2 and 7. URBM normalization improves also the performance in this case but the amount of improvement is not as much as for sigmoid transformation. If the URBM is trained with sigmoid hidden units and then the parameters are used for linear transformation of input supervectors, the performance will be worse than log sigmoid with cosine scoring but better with PLDA scoring. The use of ReLU for training the URBM and linear function for transformation, keeps the performance as good as log sigmoid with cosine scoring and improves the PLDA results obtained with sigmoid URBM and linear transformation. URBM trained with VReLU improves the PLDA results slightly more. We will show later that VReLU works better than ReLU on unseen evaluation set with both cosine and PLDA scoring.

Table 3 compares the performance of GMM–RBM vectors, which are obtained with URBMs trained with ReLU and VReLU, with traditional i-vectors on the evaluation set. The use of proposed VReLU shows better performance than the use of ReLU in both cosine and PLDA scoring. This fact implies that the variable threshold τ in VReLU has increased the generalization power of URBM in addition to the correction of the histograms. As in this table, the performance of the best GMM–RBM vectors is comparable to that of i-vectors for both cosine and PLDA scoring. This is a significant achievement since the computational load of GMM–RBM vector extraction is much less than the traditional i-vector extraction as discussed in Section 3.4. At the end, the best results are achieved with score fusion of i-vectors and GMM–RBM vectors which shows about 7–7.5% and 4–6.5% relative improvements in terms of EER and minDCF, respectively, compared to i-vectors. For score fusion, BOSARIS toolkit (Brummer and Villiers, 2011) is used. The fusion weights are trained on the development set.

Table 2
The effect of the type of the hidden units during training the URBM and the transformation function for the extraction of GMM–RBM vectors. Results are obtained on the **development** database with vectors of dimension 400. VReLU refers to the proposed variable ReLU.

Hidden units	Transformation	Cosine		PLDA	
		EER (%)	minDCF	EER (%)	minDCF
Sigmoid	Sigmoid	13.55	0.0570	11.05	0.0517
	Sigmoid (normalized URBM)	8.67	0.0407	6.08	0.0338
	Log sigmoid	8.08	0.0383	6.51	0.0316
	Log sigmoid (normalized URBM)	7.85	0.0366	6.28	0.0317
	Linear	8.24	0.0382	5.86	0.0317
ReLU	Linear	7.82	0.0372	5.58	0.0305
VReLU	Linear	7.82	0.0373	5.52	0.0297

Table 3
Performance comparison of proposed GMM−RBM vectors and conventional i-vectors on the **evaluation** set core test condition-common 5 of NIST 2010 SRE. GMM−RBM vectors and i-vectors are of a same size of 400.

		Cosine		PLDA	
		EER (%)	minDCF	EER (%)	minDCF
[1]	i-Vector	6.270	0.05450	4.096	0.04993
[2]	GMM−RBM vector (trained with ReLU)	6.638	0.06228	4.517	0.05085
[3]	GMM−RBM vector (trained with VReLU)	6.497	0.06099	3.907	0.05184
Fusion [1] and [3]		**5.791**	**0.05238**	**3.814**	**0.04673**

5. Conclusion

We have presented in this work a new vector representation of speech for text-independent speaker recognition. GMM supervectors have been transformed by a Universal RBM (URBM) to lower dimensional vectors, referred to as GMM−RBM vectors. The role of URBM has been to learn the total speaker and session variability among background GMM supervectors. The use of different hidden units for training of URBM and different transformation functions for vector extraction are investigated. A variant of linear rectified units (ReLU), which is referred to as variable ReLU (VReLU), is proposed. The variable threshold defined in these units corrects the histograms of GMM−RBM vectors and leads to higher generalization power of URBM. The experimental results on the core test-common condition 5 of NIST 2010 SRE show that the performance of GMM−RBM vectors is comparable with that of traditional i-vectors with both cosine and PLDA scoring but with much less computational load. Moreover, the best results are obtained by score fusion of GMM−RBM vectors and i-vectors.

Acknowledgments

This work has been funded by the Spanish project DeepVoice (TEC2015-69266-P) and the European project CAMOMILE (PCIN-2013-067). We would like to thank Miquel India for his help in the extraction of the traditional i-vectors for the NIST SRE 2010 data.

References

Bahari, M.H., McLaren, M.L., Hamme, H.v., Leeuwen, D.A.v., 2012. Age estimation from telephone speech using i-vectors. In: Proceedings of the 2012 Annual Conference of the International Speech Communication Association (Interspeech).

Brummer, N., Villiers, E., 2011. BOSARIS toolkit user guide: theory, algorithms and code for binary classifier score processing. [Online]. Available: https://sites.google.com/site/bosaristoolkit/.

Campbell, W.M., 2014. Using deep belief networks for vector-based speaker recognition. In: Proceedings of the 2014 Annual Conference of the International Speech Communication Association (Interspeech), pp. 676–680.

Dahl, G., Yu, D., Deng, L., Acero, A., 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Trans. Audio Speech Lang. Process. 20 (1), 30–42. doi: 10.1109/TASL.2011.2134090.

Dehak, N., Dehak, R., Glass, J., Reynolds, D., Kenny, P., 2010. Cosine similarity scoring without score normalization techniques. In: Proceedings of the 2010 Speaker and Language Recognition Workshop (Odyssey).

Dehak, N., Kenny, P., Dehak, R., Dumouchel, P., Ouellet, P., 2011a. Front-end factor analysis for speaker verification. IEEE Trans. Audio Speech Lang. Process. 19 (4), 788–798. doi: 10.1109/TASL.2010.2064307.

Dehak, N., Torres-Carrasquillo, P., Reynolds, D., Dehak, R., 2011b. Language recognition via i-vectors and dimensionality reduction. In: Proceedings of the 2011 Annual Conference of the International Speech Communication Association (Interspeech). Citeseer, pp. 857–860.

Garcia-Romero, D., Zhang, X., McCree, A., Povey, D., 2014. Improving speaker recognition performance in the domain adaptation challenge using deep neural networks. In: Proceedings of the 2014 IEEE Spoken Language Technology Workshop (SLT), pp. 378–383.

Ghahabi, O., Hernando, J., 2014a. Deep belief networks for i-vector based speaker recognition. In: Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1700–1704.

Ghahabi, O., Hernando, J., 2014b. i-vector modeling with deep belief networks for multi-session speaker recognition. In: Proceedings of the 2014 Speaker and Language Recognition Workshop (Odyssey), pp. 305–310.

Ghahabi, O., Hernando, J., 2015. Restricted Boltzmann machine supervectors for speaker recognition. In: Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4804–4808.

Ghahabi, O., Hernando, J., 2017. Deep learning backend for single and multisession i-vector speaker recognition. IEEE/ACM Trans. Audio Speech Lang. Process. 25 (4), 807–817. doi: 10.1109/TASLP.2017.2661705.

Hinton, G., 2012. A practical guide to training restricted Boltzmann machines. Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 599–619.

Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., 2012. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process. Mag. 29 (6), 82–97.

Hinton, G., Osindero, S., Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18 (7), 1527–1554. doi: 10.1162/neco.2006.18.7.1527.

Hinton, G., Salakhutdinov, R., 2006. Reducing the dimensionality of data with neural networks. Science 313 (5786), 504–507. doi: 10.1126/science.1127647.

Kenny, P., 2010. Bayesian speaker verification with heavy tailed priors. In: Proceedings of the 2010 Speaker and Language Recognition Workshop (Odyssey).

Kenny, P., Gupta, V., Stafylakis, T., Ouellet, P., Alam, J., 2014. Deep neural networks for extracting Baum−Welch statistics for speaker recognition. In: Proceedings of the 2014 Speaker and Language Recognition Workshop (Odyssey), pp. 293–298.

Larcher, A., Bonastre, J.-F., Fauve, B., Lee, K., Lvy, C., Li, H., Mason, J., Parfait, J.-Y., 2013. ALIZE 3.0 open source toolkit for state-of-the-art speaker recognition. In: Proceedings of the 2013 Annual Conference of the International Speech Communication Association (Interspeech), pp. 2768–2771.

Lei, Y., Scheffer, N., Ferre, L., Mclaren, M., 2014. A novel scheme for speaker recognition using a phonetically-aware deep neural network. In: Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

Liu, Y., Qian, Y., Chen, N., Fu, T., Zhang, Y., Yu, K., 2015. Deep feature for text-dependent speaker verification. Speech Commun. 73, 1–13.

Lozano-Diez, A., Silnova, A., Matejka, P., Glembek, O., Plchot, O., Pesan, J., Burget, L., Gonzalez-Rodriguez, J., 2016. Analysis and optimization of bottleneck features for speaker recognition. In: Proceedings of the 2016 Speaker and Language Recognition Workshop (Odyssey), pp. 352–357.

Mclaren, M., Lei, Y., Ferre, L., 2015. Advances in deep neural network approaches to speaker recognition. In: Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

Mohamed, A., Dahl, G., Hinton, G., 2012. Acoustic modeling using deep belief networks. IEEE Trans. Audio Speech Lang. Process. 20 (1), 14–22. doi: 10.1109/TASL.2011.2109382.

Mohamed, A., Yu, D., Deng, L., 2010. Investigation of full-sequence training of deep belief networks for speech recognition. In: Proceedings of the 2010 Annual Conference of the International Speech Communication Association (Interspeech), pp. 2846–2849.

Nadeu, C., Macho, D., Hernando, J., 2001. Time and frequency filtering of filter-bank energies for robust HMM speech recognition. Speech Commun. 34 (1−2), 93–114. doi: 10.1016/S0167-6393(00)00048-0.

NIST, 2006. The NIST year 2006 speaker recognition evaluation plan. [Online]. Available: http://www.nist.gov/speech/tests/spk/2006/index.htm.

NIST, 2010. The NIST year 2010 speaker recognition evaluation plan. [Online]. Available: https://www.nist.gov/itl/iad/mig/speaker_recognition_evaluation_2010.

Novoselov, S., Pekhovsky, T., Simonchik, K., Shulipa, A., 2014. RBM-PLDA subsystem for the NIST i-vector challenge. In: Proceedings of the 2014 Annual Conference of the International Speech Communication Association (Interspeech), pp. 378–382.

Pelecanos, J., Sridharan, S., 2001. Feature warping for robust speaker verification. In: Proceedings of the 2001 Speaker and Language Recognition Workshop (Odyssey). Crete, Greece, pp. 213–218.

Prince, S., Elder, J., 2007. Probabilistic linear discriminant analysis for inferences about identity. In: Proceedings of the Eleventh IEEE International Conference on Computer Vision, (ICCV 2007).

Richardson, F., Reynolds, D., Dehak, N., 2015. Deep neural network approaches to speaker and language recognition. IEEE Signal Process. Lett. 22 (10), 1671–1675.

Safari, P., Ghahabi, O., Hernando, J., 2015. Feature classification by means of deep belief networks for speaker recognition. In: Proceedings of the 2015 European Signal Processing Conference (EUSIPCO), pp. 2162–2166.

Safari, P., Ghahabi, O., Hernando, J., 2016. From features to speaker vectors by means of restricted Boltzmann machine adaptation. In: Proceedings of the 2016 Speaker and Language Recognition Workshop (Odyssey), pp. 366–371.

Senior, A., Sak, H., Shafran, I., 2015. Context dependent phone models For LSTM RNN acoustic modelling. In: Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4585–4589.

Senoussaoui, M., Dehak, N., Kenny, P., Dehak, R., Dumouchel, P., 2012. First attempt of Boltzmann machines for speaker verification. In: Proceedings of the 2012 Speaker and Language Recognition Workshop (Odyssey).

Stafylakis, T., Kenny, P., Senoussaoui, M., Dumouchel, P., 2012a. PLDA using gaussian restricted Boltzmann machines with application to speaker verification. In: Proceedings of the 2012 Annual Conference of the International Speech Communication Association (Interspeech).

Stafylakis, T., Kenny, P., Senoussaoui, M., Dumouchel, P., 2012b. Preliminary investigation of Boltzmann machine classifiers for speaker recognition. In: Proceedings of the 2012 Speaker and Language Recognition Workshop (Odyssey).

Variani, E., Lei, X., McDermott, E., Lopez Moreno, I., Gonzalez-Dominguez, J., 2014. Deep neural networks for small footprint text-dependent speaker verification. In: Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4052–4056.

Vasilakakis, V., Cumani, S., Laface, P., 2013. Speaker recognition by means of deep belief networks. In: Proceedings of the 2013 Biometric Technologies in Forensic Science, pp. 52–57.

Xia, R., Liu, Y., 2012. Using i-vector space model for emotion recognition. In: Proceedings of the 2012 Annual Conference of the International Speech Communication Association (Interspeech), pp. 2227–2230.

Xiang, B., Chaudhari, U.V., Navrtil, J., Ramaswamy, G.N., Gopinath, R.A., 2002. Short-time Gaussianization for robust speaker verification. In: Proceedings of the 2002 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 681–684.

Omid Ghahabi received the M.Sc. degree in electrical engineering from Shahid Beheshti University, Tehran, Iran, in 2009. He is currently working toward the Ph.D. degree at the Technical University of Catalonia (UPC), Barcelona, Spain. From 2009 to 2011, he was in the Speech Processing Group, Research Center for Intelligent Signal Processing, Tehran, Iran. Between 2011 and 2016, he was a Researcher in the Speech Processing Group, Signal Theory and Communications Department, UPC. Since late 2016, he has been in the EML European Media Laboratory GmbH, Heidelberg, Germany, as a Speech Technologist. His research interests include speaker and language recognition, speaker diarization, speech signal processing, and deep learning. He is the author and coauthor of several journal and conference papers on these topics. He is a member of the Research Center for Language and Speech Technologies and Applications, Barcelona, Spain.

Javier Hernando received the M.S. and Ph.D. degrees in telecommunication engineering from the Technical University of Catalonia (UPC), Barcelona, Spain, in 1988 and 1993, respectively. Since 1988, he has been in the Department of Signal Theory and Communications, UPC, where he is currently a Full Professor and the Director of the Research Center for Language and Speech. During the academic year 2002–2003, he was a Visiting Researcher in the Panasonic Speech Technology Laboratory, Santa Barbara, CA, USA. He has led the UPC team in several European, Spanish, and Catalan projects. His research interests include robust speech analysis, speech recognition, speaker verification and localization, oral dialogue, and multimodal interfaces. He is the author or coauthor of about 200 publications in book chapters, review articles, and conference papers on these topics. He received the 1993 Extraordinary Ph.D. Award of UPC.