Odyssey 2014:
The Speaker and Language Recognition Workshop
16-19 June 2014, Joensuu, Finland

i-Vector Modeling with Deep Belief Networks for Multi-Session Speaker Recognition

Omid Ghahabi and Javier Hernando

TALP Research Center, Department of Signal Theory and Communications
Universitat Politecnica de Catalunya - BarcelonaTech, Spain
{omid.ghahabi,javier.hernando}@upc.edu

Abstract

In this paper we propose an impostor selection method for a Deep Belief Network (DBN) based system which models i-vectors in a multi-session speaker verification task. In the proposed method, instead of choosing a fixed number of most informative impostors, a threshold is defined according to the frequencies of impostors. The selected impostors are then clustered and the centroids are considered as the final impostors for target speakers. The system first trains each target speaker unsupervisingly by an adaptation method and then models discriminatively each target speaker using the impostor centroids and target i-vectors. The evaluation is performed on the NIST 2014 i-vector challenge database and it is shown that the proposed DBN-based system achieves 23% relative improvement of minDCF over the baseline system in the challenge.

1. Introduction

Speaker recognition based on the i-vector framework [1] is widely accepted as a state-of-the-art in this field. An i-vector is a compact representation of the speaker useful information which is obtained over an effective factor analysis method [1]. The i-vectors are further post-processed to compensate undesired speaker and session variabilities [1][2][3][4]. However, the main focus has been on the single session speaker verification (eg. [1] [2][5][6]). In this sense that only one speech utterance is available per each target speaker. The few available research works about multi-session speaker verification are mostly using either the average i-vectors obtained over the session i-vectors or the combination of scores obtained on each individual session i-vector [7][8] [9].

The National Institute of Standard and Technologies (NIST) organizes some speaker recognition evaluations to encourage research groups to develop more efficient systems. The most recent challenge is planned for modeling i-vectors in a multi-session enrollment task [10]. The good point of the challenge is that i-vectors are given directly, instead of speech signals. Therefore, the front-end will be the same for all participating systems and the focus will be mostly on the modeling part.

Acoustic modeling using Deep Belief Networks (DBN) has been recently shown to be effective in speech recognition [11][12][13] [14]. However, few attempts using only Restricted Boltzmann Machines (RBM) [15][16], generative DBNs [17], or discriminative ones [18] have been carried out in speaker recognition area. The most recent research work [18] has shown that using DBNs is effective in a single session i-vector modeling. DBNs are originally generative network models which can be trained by a greedy layer-wise algorithm using RBMs [19][20]. However, by adding a top label layer and using a standard backpropagation algorithm, these generative DBNs can be converted to discriminative ones what is called often a pre-trained discriminative network [20][13].

In this paper we will use DBNs to model multi-session target i-vectors. As in our previous work [18] we will take the advantage of unsupervised learning to model a global DBN to be used in an adaptation process and the advantage of supervised learning to model each target speaker discriminatively. As more i-vector samples are available per each target speaker in this case and each of them may be recorded from different session, DBNs will capture more speaker and session variabilities from the input data and will work better than in the single session task. Moreover, the impostor selection method proposed in [18] will be modified to some extent. Instead of fixing a threshold on the number of most informative impostors (e.g., 500, 1000, etc.), the threshold will be fixed on the impostor frequency values.

The rest of the paper is organized as follows. Sections 2 and 3 review, respectively, the i-vector framework and a background on DBN. The general description of the DBN-based system is given in Section 4. Section 5 describes the proposed impostor selection method. Section 6 presents the experimental setup and results. And section 7 concludes the paper.

2. i-Vector Extraction

This section has a brief overview on the i-vector framework developed in [1]. Given the centralized Baum-Welch statistics from all available speech utterances, the low rank total variability matrix ($\mathbf{T}$) is trained in an iterative process. This matrix tries to capture all kinds of variabilities, including speaker and session variabilities,

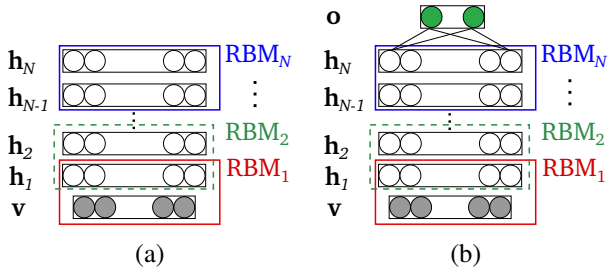

Figure 1: Generative (a) and discriminative (b) DBNs.

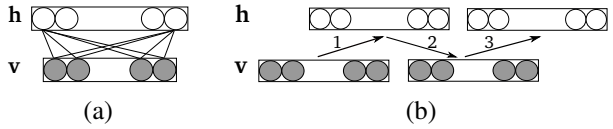

Figure 2: RBM (a) and RBM training (b).

appeared in training utterances. The training process assumes that an utterance can be represented by the Gaussian Mixture Model (GMM) mean supervector,

$$\mathbf{m} = \mathbf{m}_u + \mathbf{T}\boldsymbol{\omega} \tag{1}$$

where $\mathbf{m}_u$ is the speaker- and session-independent mean supervector from the Universal Background Model (UBM), and $\boldsymbol{\omega}$ is a low rank vector referred to as the identity vector or i-vector. The supervector $\mathbf{m}$ is assumed to be normally distributed with the mean $\mathbf{m}_u$ and the covariance $\mathbf{T}\mathbf{T}^t$, and the i-vectors have a standard normal distribution $\mathcal{N}(0,1)$. More details can be found in [1].

3. Deep Belief Networks

DBNs are originally probabilistic generative models with multiple layers of stochastic hidden units above a layer of visible variables which represent a data vector (Fig. 1a). The network parameters are trained using an efficient unsupervised algorithm which is equivalent to training each two adjacent layers as a Restricted Boltzmann Machine (RBM) (Figs. 2a and 1a) [20]. RBMs are constructed from a layer of binary stochastic hidden units and a layer of stochastic visible units which will be either Bernoulli or Gaussian distributed conditional on the hidden units. Training an RBM is based on stochastic gradient descent on the negative log likelihood l defined in [20][13],

$$\Delta w_{ij} = -\alpha\left(\frac{\partial l}{\partial w_{ij}}\right) \tag{2}$$

where α is the learning rate and w_{ij} represents the weight between the visible unit i and the hidden unit j. As computing the gradient is infeasible in this case [20][13], it is approximated by the Contrastive Divergence (CD) algorithm [19][20]. Since a full CD algorithm is computationally expensive, it is further approximated in the following three steps (Fig. 2b) and is called CD-1 [20][13]. After initializing the connection weights with very small normal-distributed random numbers ($\mathcal{N}(0, 0.01)$) and setting the bias values to zero, hidden states ($\mathbf{h}$) are computed with the posterior probability distribution,

$$p(h_j = 1|\mathbf{v}, \theta) = \sigma(a_j + \sum_{i=1}^{V} w_{ij} v_i) \tag{3}$$

where $\theta = (\mathbf{w}, \mathbf{b}, \mathbf{a})$ refers to the RBM parameters, V is the number of visible units, b_i and a_j are respectively the bias terms of visible unit i and the hidden unit j, and $\sigma(x) = (1 + e^{-x})^{-1}$ is the sigmoid function.

Then given $\mathbf{h}$, the Bernoulli distributed visible states are reconstructed in the same manner as in eq. (3) and the real-valued Gaussian ones are reconstructed by,

$$p(v_i|\mathbf{h}, \theta) = \mathcal{N}(b_i + \sum_{j=1}^{H} w_{ij} h_j, 1) \tag{4}$$

where $\mathcal{N}(\mu, \sigma^2)$ is a Normal function and H is the number of hidden units. In the third step, given the reconstructed data ($\mathbf{v}$) and the eq. (3), the new values for the hidden states are computed.

Now, the negative gradient in eq. (2) is approximated as follows,

$$-\frac{\partial l}{\partial w_{ij}} \approx \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon} \tag{5}$$

where $\langle . \rangle_{data}$ and $\langle . \rangle_{recon}$ denote the expectations when the hidden state values are driven respectively from the input visible data and the reconstructed data. The biases are updated in a similar way.

It is possible to perform the above parameter update after processing each training example, but it is often more efficient to divide the whole input data (batch) into smaller size batches (minibatch) and to do the parameter update by an average over each minibatch. The parameter updating procedure is iterated when the whole available input data are processed. Each iteration is called an epoch. More theoretical and practical details can be found in [19][20][21].

When the unsupervised learning is finished, by adding a label layer on top of the network and doing a supervised backpropagation training, it can be converted to a discriminative model (Fig. 1b). In other words, unsupervised learning can be considered as a pre-training for the supervised stage. It has been shown [20] that this unsupervised pre-training can set the weights of the network to be closer to a good solution than random initialization and, therefore, avoids local minima when using supervised gradient descent.

4. i-Vector Modeling Using DBN

The main idea is to model discriminatively the target and impostor i-vectors by a DBN structure. The structure which was proposed for the first time in a single session enrollment task [18] will be used also in this paper to model speakers with more target i-vectors available. In this case it is expected to have more accurate

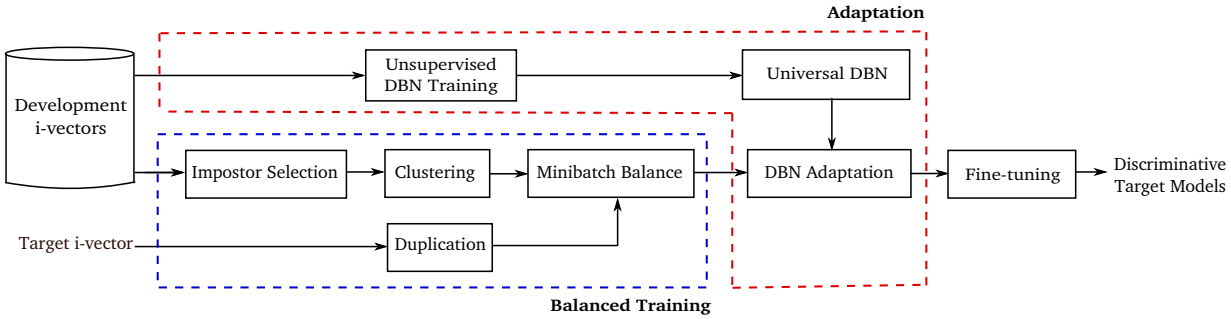

Figure 3: Block-diagram of the proposed method.

models as DBNs are being shown more positive samples and, therefore, more speaker and session variabilities. In this section, we describe briefly the whole structure used, and in the next section focus on the impostor selection method which is the main new contribution of this paper. As illustrated in Fig. 3, the DBN structure is composed of three main parts namely balanced training, adaptation, and fine-tuning.

Like other discriminative methods, DBNs need also balanced positive and negative input data to achieve their best results. The balanced training part in the block diagram (Fig. 3) tries to use the information of all available impostors and decrease their population in a reasonable way. The decreasing is carried out in two steps, selecting the most informative ones and clustering. In [18] a simple and effective selection method is proposed. First, the n closest impostors to each target speaker are chosen according to their cosine distances. Then the closest impostors are accumulated over all target speakers and the k top ranked impostors are selected according to the number of times they are appeared in the accumulated set of impostors. In other words, the k impostors which are statistically closer to all target speakers are selected by this method. The selected impostors are clustered finally by the k-means algorithm using the cosine distance criteria. On the contrary, as there is only one target i-vector in the single session task, they are duplicated as many times as the number of final impostor clusters. However, when more than one positive sample are available per each target speaker, we will choose the number of impostor clusters in each minibatch the same as the number of available positive samples to make the training balanced. Hence, if the number of minibatches is set to three, for instance, and the number of positive samples per each speaker is five, the total number of impostor clusters will be 15. Actually, in each minibatch we will show the network the same positive samples as in other minibatches but different negative ones.

DBNs have the ability to be trained unsupervisingly [20][19] contrary to conventional neural networks that need labeled data to be trained. Hence, a global model called Universal DBN (UDBN) [18] is trained by feeding many i-vectors from development background data. The training is carried out layer by layer using RBMs as described in section 3. UDBN parameters are adapted to the new data of each speaker including both target and impostor samples obtained in the balanced training part of Fig. 3. The adaptation is carried out by pre-training each network initialized by the UDBN parameters. It is shown [18] that the adaptation process outperforms both random and pre-training initializations.

Once the adaptation process is completed, a label layer is added on the top of the network and the stochastic gradient descent backpropagation is carried out as the fine-tuning process. The softmax will be the activation function of the top label layer. To minimize the negative effect of using random numbers used for initializing the top layer parameters, a pseudo pre-training process is performed by only one layer error backpropagating for a few iterations before a full backpropagation is carried out. If the input labels in the training phase are chosen as $(l_1 = 1, l_2 = 0)$ and $(l_1 = 0, l_2 = 1)$ for target and impostor i-vectors respectively, the final output score in the testing phase will be computed in a Log Likelihood Ratio (LLR) form as follows,

$$LLR = log(o_1) - log(o_2) \quad (6)$$

where (o_1, o_2) represents the outputs of the top layer. LLR computation helps to gaussianize the true and false score distributions which can be useful for score fusion.

5. Impostor Selection

The idea is to design a more accurate and robust impostor selection method in our system. The base of the proposed method is the same as that in [18] in which the statistically closest impostors to all available target speakers are selected as the most informative ones. However, instead of choosing a round number of top ranked impostors at the end (e.g., 500, 1000, etc.), they are selected according to a threshold (T in Fig. 4a) which will be applied on the impostor frequencies.

The whole selection procedure is as follows. At first, the mean i-vector of each target speaker, in the multi session task, is scored against all available impostor i-vectors in the development set according to their cosine distances. Then the n closest impostors to each given target speaker are kept in the set H. Therefore, each impostor may appear in H several times. The number of

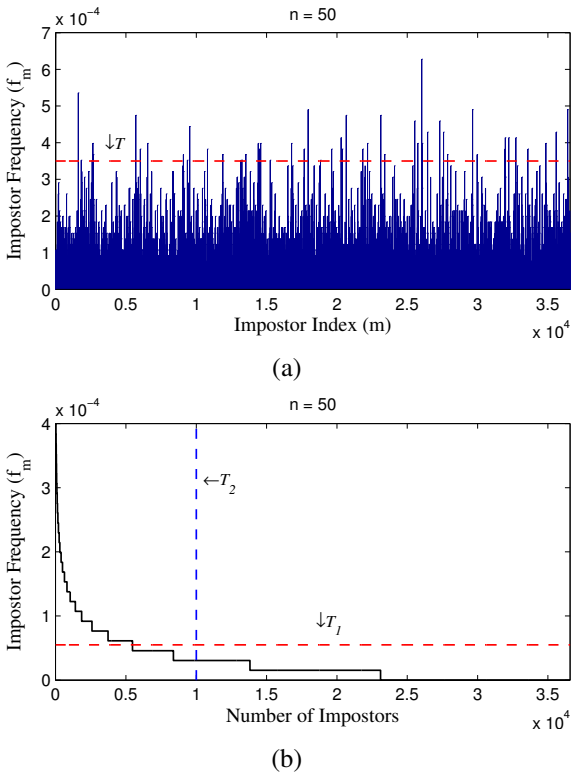

Figure 4: Impostor selection based on swiping a threshold on the impostor frequency values. (a) non-ordered, and (b) ordered impostors.

times that each impostor appears in the set H is denoted l_m. If l_m is normalized by the total number of samples in H, we will have

$$f_m = \frac{l_m}{\sum_{m=1}^{M} l_m} = \frac{l_m}{n \times K}, \quad (7)$$

where f_m is the relative frequency of impostor m. M and K are, respectively, the total number of impostor and target speakers. Actually, working with these frequencies will be more robust as they are normalized by the parameter n and the total number of target speakers. Hence, the change of the target database or the parameter n will not affect the impostor frequencies. Consequently, the threshold based on which we select the impostors will be more robust.

Figure 4a shows the frequencies for all the available impostors in the development set for $n = 50$. The more informative impostors have the higher frequency values in comparison to other impostors. As shown in the figure, by swiping a threshold we can select the most statistically important impostors. In other words, only those impostors which have the frequency values higher than the threshold will be selected. If the impostors are ordered according to their frequencies (Fig. 4b), it can be seen that many of them have the same frequency values. Actually, the lower value for parameter n the more number of impostors with the same frequency. As there is no priority among the impostors with the same frequency, it makes more sense to choose the impostors by fixing the threshold on frequencies (T_1 in Fig. 4b), instead of on the number of impostors (T_2 in Fig. 4b). If we select a round number of impostors (like T_2 in Fig. 4b) as it is carried out in [22][23][18], we have actually discarded the rest of the impostors with the same frequency. We will choose the best n and T in section 6.

The selected impostors will be further clustered by the k-means algorithm and the cluster centroids will be considered as the final impostors (Fig. 3). The selected impostors, before clustering, are statistically close to all of the target speakers and, therefore, will be target-independent. We will see in section 6 that if we pool the target-independent impostors with target-dependent ones (the n closest impostors to each target speaker in this case) as it is proposed in [23], we will achieve better results. However, it would be more computationally expensive as we need to do clustering for each target speaker independently.

6. Multi-Session Experiments

The details of the database, the baseline and the DBN-based setups, and the obtained results are given in this section.

6.1. Baseline and Database

The experiments are carried out on the NIST 2014 i-vector challenge [10]. In this challenge contrary to other previous NIST evaluations, i-vectors are provided instead of speech signals. The i-vectors are computed from conventional telephone speech recordings in the SRE 2004 to 2012. The durations of speech utterances used to obtain i-vectors are different. They are sampled from a normal distribution with a mean of 40 s. The length of each i-vector is 600. Three sets of i-vectors are provided: unlabeled development, model, and test. The amounts of i-vectors in each set are respectively 36,572, 6,530, and 9,634. The number of target models is 1,306 and for each of them five i-vectors are available. Each model will be scored against all the test i-vectors and, therefore, 12,582,004 trials will be reported. Among all trials, 40% (progress subset) will be scored by NIST as a feedback to develop the system and 60% (evaluation subset) will be reserved for the final official evaluation. The performance is evaluated using a new Decision Cost Function (DCF) defined by NIST [10],

$$DCF(t) = \frac{\#Miss(t)}{\#Targets} + 100 \times \frac{\#FalseAlarm(t)}{\#NonTargets} \quad (8)$$

where t refers to the threshold for which the DCF is being computed. The minimum DCF obtained over all thresholds will be the official system score.

In the baseline system, average i-vectors obtained over the available i-vectors for each target speaker are scored against all test i-vectors using cosine distance classifier. However, before averaging and scoring some post-

processing is carried out on i-vectors. The global mean and covariance are computed using unlabeled development data. All i-vectors are centered and whitened based on the global mean and covariance. Then the resulting i-vectors are length normalized. Length normalization is applied again on the average i-vectors obtained for each target speaker.

It is worth noting that, although NIST 2014 i-vector challenge database has been used in this paper, one of the rules in this challenge has not exactly followed in our experiments, in particular the one which does not allow the use of evaluation data for impostor modeling [10]. Actually, in these experiments the i-vectors of both progress and evaluation subsets have been considered for impostor selection.

6.2. DBN-based Setup

As in [18] DBNs with only one hidden layer are explored in this paper. The size of hidden layer is set to 400. Each minibatch will include five impostor centroids and five target samples. The impostor centroids in each minibatch are different than those in other ones, but they share the same target samples. The number of minibatches is set to three and, therefore, we will have 15 impostor centroids in total. The unlabeled development i-vectors provided by NIST are used for impostor selection. UDBN is trained with the same development i-vectors as in the impostor database. As the input i-vectors are real-valued normal distributed, a Gaussian-Bernoulli RBM [21][13] is employed. The learning rate (α), the number of epochs (NofE), and the minibatch size are set respectively to 0.02, 50, and 100 for UDBN training. A fixed momentum of 0.9 and a weight decay of 2×10^{-4} are also considered.

The adaptation process is carried out with $\alpha = 0.03$ and NofE=25. To decrease the probability of overfitting during the adaptation, it is performed on each minibatch separately and then the obtained network parameters are averaged. The softmax connection weights are initialized by $\mathcal{N}(0, 0.01)$ and pre-trained with $\alpha = 1$ and NofE=15 before the whole backpropagation is performed. The momentum is started by 0.4 and is scaled up by 0.1 after each epoch (up to 0.9). The whole backpropagation is then carried out with $\alpha = 1$, NofE=30, and a fixed momentum of 0.9. The weight decay for both top layer pre-tarining and the whole backpropagation is set to 0.0014.

6.3. Results

Figure 5 illustrates the variability of minDCF obtained by eq. 8 in terms of the two parameters n and T defined in sec. 5. The figure shows that the best result is obtained when $n = 100$ and $T = 0.5 \times 10^{-4}$. Table 1 compares the best results obtained by the proposed DBN-based system with the baseline. As it can be seen in this table pooling the target-independent and -dependent impostors achieves better results although it is computationally expensive. And the overall performance of the DBN-based system is notable (23% relative improvement) in comparison to the baseline.

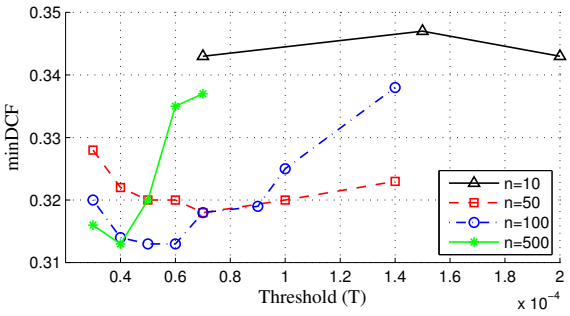

Figure 5: Determination of the parameters n and T defined in sec. 5 for impostor selection.

Table 1: Performance comparison of the DBN-based system with the baseline. The results are obtained on the NIST 2014 i-vector challenge. "dep" and "indep" stand for "dependent" and "independent", respectively.

System	Impostors	minDCF
Baseline	-	0.386
DBN-based	Target-indep	0.311
DBN-based	Target-indep + Target-dep	0.298

7. Conclusion

The authors proposed an impostor selection method which used in a Deep belief Network (DBN) system for multi-session i-vector speaker verification. The availability of more i-vector samples per each target speaker helped DBNs to capture more speaker and session variabilities from the input data in comparison to the single session task. The final discriminative DBN models showed a considerable performance in comparison to the conventional baseline system.

8. Acknowledgement

This work has been funded by the Spanish project SARAI (TEC2010-21040-C02-01).

9. References

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

[2] S.J.D. Prince and J.H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, 2007.

[3] P. Kenny, "Bayesian speaker verification with heavy tailed priors," in *Speaker and Language Recognition Workshop (IEEE Odyssey)*, 2010.

[4] N. Brummer and E. Villiers, "The speaker partitioning problem," in *Proceedings of the Odyssey Speaker and Language Recognition Workshop, Brno, Czech Republic*, 2010.

[5] O. Barkan and H. Aronowitz, "Diffusion maps for PLDA-based speaker verification," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013, pp. 7639–7643.

[6] L. He and J. Liu, "I-matrix for text-independent speaker recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013, pp. 7194–7198.

[7] G. Liu, T. Hasan, H. Boril, and J.H.L. Hansen, "An investigation on back-end for speaker recognition in multi-session enrollment," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013, pp. 7755–7759.

[8] A. Larcher, K. Lee, B. Ma, and H. Li, "Phonetically-constrained PLDA modeling for text-dependent speaker verification with multiple short utterances," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013, pp. 7673–7677.

[9] A. Larcher, J-F. Bonastre, B. Fauve, K. Lee, C. Lvy, H. Li, J. Mason, and J-Y. Parfait, "ALIZE 3.0 open source toolkit for state-of-the-art speaker recognition," in *Proc. Interspeech*, 2013, pp. 2768–2771.

[10] "The 2013-2014 speaker recognition i-vector machine learning challenge," 2014.

[11] A. Mohamed, G.E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, Jan. 2012.

[12] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, and T. Sainath, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, 2012.

[13] G.E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, Jan. 2012.

[14] A. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *Proc. Interspeech*, 2010, p. 28462849.

[15] M. Senoussaoui, N. Dehak, P. Kenny, R. Dehak, and P. Dumouchel, "First attempt of boltzmann machines for speaker verification," in *Odyssey 2012-The Speaker and Language Recognition Workshop*, 2012.

[16] T. Stafylakis, P. Kenny, M. Senoussaoui, and P. Dumouchel, "Preliminary investigation of boltzmann machine classifiers for speaker recognition," in *Odyssey 2012-The Speaker and Language Recognition Workshop*, 2012.

[17] V. Vasilakakis, S. Cumani, and P. Laface, "Speaker recognition by means of deep belief networks," in *Biometric Technologies in Forensic Science*, 2013.

[18] O. Ghahabi and J. Hernando, "Deep belief networks for i-vector based speaker recognition," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014.

[19] G.E. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, July 2006.

[20] G.E. Hinton, S. Osindero, and Y-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, May 2006.

[21] G.E. Hinton, "A practical guide to training restricted boltzmann machines," in *Neural Networks: Tricks of the Trade*, number 7700 in Lecture Notes in Computer Science, pp. 599–619. Springer Berlin Heidelberg, Jan. 2012.

[22] M. McLaren, B. Baker, R. Vogt, and S. Sridharan, "Improved SVM speaker verification through data-driven background dataset collection," in *IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009*, 2009, pp. 4041–4044.

[23] G. Liu, J-W. Suh, and J.H.L. Hansen, "A fast speaker verification with universal background support data selection," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2012, pp. 4793–4796.