Odyssey 2016
21-24 Jun 2016, Bilbao, Spain

On autoencoders in the *i*-vector space for speaker recognition

Timur Pekhovsky[1,2], *Sergey Novoselov*[1], *Aleksei Sholohov*[2,3], *Oleg Kudashev*[1]

[1]Speech Technology Center Ltd., St. Petersburg, Russia
[2]ITMO University, St. Petersburg, Russia
[3]University of Eastern Finland, Joensuu, Finland
{tim,novoselov,sholohov,kudashev}@speechpro.com

Abstract

We present the detailed empirical investigation of the speaker verification system based on denoising autoencoder (DAE) in the *i*-vector space firstly proposed in [1]. This paper includes description of this system and discusses practical issues of the system training. The aim of this investigation is to study the properties of DAE in the *i*-vector space and analyze different strategies of initialization and training of the back-end parameters. Also in this paper we propose several improvements to our system to increase the accuracy. Finally, we demonstrate potential of the proposed system in the case of domain mismatch. It achieves considerable gain in performance compared to the baseline system for the unsupervised domain adaptation scenario on the NIST 2010 SRE task.

1. Introduction

In our previous study [1] we presented a speaker verification system based on machine learning techniques originated from the deep learning [2], [3]. We proposed to use *denoising autoencoders* (DAE) for the *i*-vector based speaker verification. The *autoencoder* (AE) is a neural network that aims to reproduce its input by learning efficient data representation (encoding). The DAE is a variant of AE that attempts to reconstruct the original data point from its corrupted version. While DAEs are usually used for extracting robust features and building deep models [4], we employ them to learn a nonlinear denoising transform to reduce within-speaker variability in the *i*-vector space. In contrast to the standard procedure for learning DAEs, we do not add artificial corruptions to the features and assume that speaker representations (*i*-vectors) are already corrupted by the channel effects. Thus, proposed approach can be seen as a channel-compensation technique which aims to minimize the effect of channel in the *i*-vector speaker representations. After that a standard back-end classifier such as probabilistic linear discriminant analysis (PLDA) can be applied to the outputs of DAE. This DAE based speaker verification system achieved considerable performance improvement compared to the commonly used baseline (*i.e.* PLDA on raw *i*-vectors) [1].

In this study we continue research on denoising autoencoders for speaker verification started in [1]. We analyze the properties of the DAE system in more details than it was done in the preliminary study. In particular, we analyze the important effect of replacing the back-end parameters in the DAE system, which was not covered before. We also explore the effects of various training configurations and report system performance on several datasets.

In addition to this analysis we propose several improvements to our previous system. First, we employ dropout method [5] for DAE training. Second, we make attempts to build a deep DAE following the ideas from [6].

Also, as a part of more detailed study of the DAE system, we considered the domain mismatch problem which is important topic in the speaker recognition research. Our previous experimental setup included only homogeneous speech corpus leaving the open question about robustness of the DAE system to domain mismatch. In these experiments we followed the setup of the domain adaptation challenge (DAC) designed-by MIT-LL.

2. Detailed study of the DAE system

2.1. Datasets and experimental setup

For experiments presented in this Section and in Section 3 we used the same setup as in our previous study [1]. The training set contains telephone channel recordings from the NIST SRE 1998-2008 corpora. It includes 16618 sessions of 1763 male speakers (only English language). We also used 228 male speakers (7065 files) from the training data set of the NIST 2010 SRE as the cross-validation set for detecting the stopping point during autoencoder training. We measured speaker verification system performance according to the protocol from the NIST 2010 SRE (condition 5 extended, males, English language) [7]. All the evaluation results are presented in terms of two operating points: *equal error rate* (EER) and *minimum detection cost function* (minDCF) with probability of target trial set to 10^{-3} [7].

2.2. Description of the DAE system

Here we describe our speaker verification system in more detail than it was done in [1]. In this study we do not consider any issues in the front-end or *i*-vector extraction and focus only on data modeling in the *i*-vector space. The detailed block diagram of the DAE system is shown in Figure 1.

The key element in this diagram is the *denoising restricted Boltzmann machine* (RBM) introduced in [1]. For each speaker, this RBM models the joint distribution of the "mean" *i*-vector of the speaker and another *i*-vector corresponding to a session of the same speaker. Trained RBM can map any *i*-vector to its denoised version reducing the effect of channel. This denoising transform can be further fine-tuned using discriminative objective termed as denoising autoencoder (DAE). More details about denoising transform can be found in Section 2.2.2. In the end we pass *i*-vectors through the learned nonlinear transform and use the standard PLDA back-end for making decision about speakers' identity [1].

During our experiments we found out that *whitening* and *length normalization* (LN) [8] are critical for training RBM.

doi: 10.21437/Odyssey.2016-31

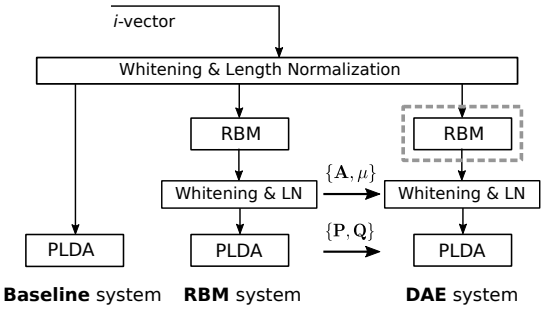

Figure 1: Block diagram of speaker recognition systems compared in our experiments. Arrows denote transferring of parameters and dashed rectangle denotes discriminative training of the encircled part.

Following the standard recipe, first, we estimate dataset mean vector $\boldsymbol{\mu}$ and the whitening matrix $\boldsymbol{A}$, then we project whitened *i*-vector on the unit hypersphere [8] (hereinafter these three steps are referred to as *normalization*). Accordingly, as shown on the diagram in Figure 1, we also use the same normalization block before DAE training.

2.2.1. Front-end

We computed 20 MFCC (including C0) with their first- and second-order derivatives using freely available Kaldi toolkit [9]. Similarly to [10], deep neural network (DNN) was used to calculate Baum-Welch statistics for the *i*-vector extraction. The outputs of the DNN correspond to the set of triphone states as well as non-speech states (noise, silence, laughing, etc.). This allowed us to use DNN outputs without any standalone voice activity detection (VAD) in an elegant way. The sum of triphone state outputs is closer to one for speech segments and closer to zero for non-speech segments. To eliminate the influence of non-speech segments to the Baum-Welch statistics it is necessary to remove components corresponding to non-speech states. Our experiments showed that such approach outperforms "hard" removal of non-speech segments according to VAD. Mean and variance normalization was done by re-normalizing Baum-Welch statistics according to formula:

$$\hat{\boldsymbol{F}}_c = (\boldsymbol{F}_c - \boldsymbol{m}N_c)\boldsymbol{\sigma}^{-1}, \quad c \in I_{tri}$$
$$\boldsymbol{m} = \frac{\sum\limits_{c \in I_{tri}} \boldsymbol{F}_c}{\sum\limits_{c \in I_{tri}} N_c}, \quad \boldsymbol{\sigma}^2 = \frac{\sum\limits_{c \in I_{tri}} \boldsymbol{S}_c}{\sum\limits_{c \in I_{tri}} N_c} - \boldsymbol{m}^2,$$

where I_{tri} is the set of DNN output indices corresponding to triphone states. N_c, $\boldsymbol{F}_c$, $\boldsymbol{S}_c$ are the 0th- 1st- and 2nd-order statistics calculated for the raw 60-dimensional features. In contrast to the commonly used approach feature normalization was applied both for MFCC and their deltas. DNN contains 2700 thiphone states and 20 non-speech states trained on the Switchboard corpus [11] using the standard recipes implemented in Kaldi. We extracted 400-dimensional *i*-vectors.

2.2.2. DAE training

The DAE training starts from generative supervised training of the denoising RBM (Figure 2, left). Similarly to [12], this RBM has binary hidden layer and Gaussian visible layer, taking a concatenation of two real-valued vectors as an input. The first vector $\boldsymbol{i}(s,h)$ is an *i*-vector extracted from the h-th session of the s-th speaker, the second vector $\boldsymbol{i}(s)$ is the average over all sessions of this speaker. $\boldsymbol{i}(s)$ can be viewed as the maximum likelihood estimate in the following model of within-speaker variability: $\boldsymbol{i}(s,h) \sim \mathcal{N}(\boldsymbol{i}(s), \boldsymbol{\Sigma}_W)$, where $\mathcal{N}(\cdot)$ is the Gaussian distribution with mean $\boldsymbol{i}(s)$ and covariance $\boldsymbol{\Sigma}_W$.

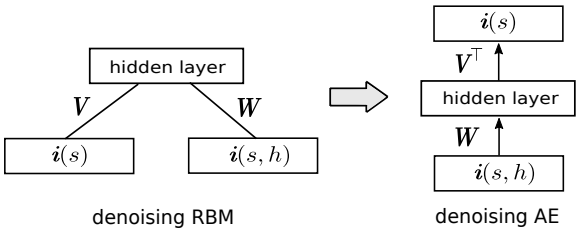

Figure 2: Learning denoising transform. $\boldsymbol{i}(s,h)$ is the *i*-vector representing h-th session of s-th speaker. $\boldsymbol{i}(s)$ is the mean *i*-vecotor for speaker s. RBM parameters are used to initialize denoising neural network.

The input layer is connected to the hidden layer by bidirectional connections represented by rectangular weight matrices $\boldsymbol{W}$ and $\boldsymbol{V}$. In our experiments the optimal size of the hidden layer was found to be 1300. We trained RBM by running Contrastive Divergence algorithm (CD-1) [13] for 20 epochs on the training set divided into small mini-batches of 20 cases.

Then we "unfold" trained RBM to form the neural network which we refer as denoising autoencoder (DAE) [14] (Figure 2, right). DAE is discriminatively trained (fine-tuned) to minimize within-speaker variability, defined in the following way:

$$\sum_s \sum_h \|\boldsymbol{i}(s) - f(\boldsymbol{i}(s,h))\|^2 \to \min, \tag{1}$$

where $f(\boldsymbol{x}) = \boldsymbol{V}^\top \sigma(\boldsymbol{W}\boldsymbol{x})$ is the denoising transform and $\sigma(\cdot)$ is the sigmoid function.

To optimize this objective we used Carl Rasmussen's `minimize.m` function [15] which implements conjugate gradient algorithm with Polak-Ribiére updates. On each iteration we computed minDCF on the cross-validation set to detect a stopping point.

2.2.3. Back-end

Given either raw or nonlinearly transformed *i*-vectors, we employ PLDA back-end to compute the scores for speaker verification. Specifically we used two-covariance model [16] which can be viewed as a special case of PLDA. In order to carry out a large number of experiments we did not use costly EM-algorithm [17, 18] to estimate the model parameters. Instead of that, as in our previous study [1], we estimated between-speaker and within-speaker covariances, respectively, according to formulas:

$$\boldsymbol{\Sigma}_B = \frac{1}{S}\sum_{s=1}^{S} (\boldsymbol{i}(s) - \boldsymbol{\mu})(\boldsymbol{i}(s) - \boldsymbol{\mu})^\top, \tag{2}$$

$$\boldsymbol{\Sigma}_W = \frac{1}{S}\sum_{s=1}^{S} \frac{1}{H(s)} \sum_{h=1}^{H(s)} (\boldsymbol{i}(s,h) - \boldsymbol{i}(s))(\boldsymbol{i}(s,h) - \boldsymbol{i}(s))^\top, \tag{3}$$

where $\boldsymbol{i}(s,h)$, $\boldsymbol{i}(s)$ were defined in the Section 2.2.2, $\boldsymbol{\mu}$ is the dataset mean, S is the number of speakers in the training set and $H(s)$ is the number of sessions of s-th speaker.

Given a pair of *i*-vectors $\boldsymbol{i}_1$ and $\boldsymbol{i}_2$, assuming zero mean and skipping the scalar term, the commonly used PLDA verification score can be written as [8]:

$$\text{score}(\boldsymbol{i}_1, \boldsymbol{i}_2) = \boldsymbol{i}_1 \boldsymbol{P} \boldsymbol{i}_2 + \boldsymbol{i}_2 \boldsymbol{P} \boldsymbol{i}_1 + \boldsymbol{i}_1 \boldsymbol{Q} \boldsymbol{i}_1 + \boldsymbol{i}_2 \boldsymbol{Q} \boldsymbol{i}_2,$$

where square matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ can be expressed in terms of (2) and (3) [8].

2.2.4. Replacing back-end

During our experiments we found out that we get the best performance for the DAE system if we use the set of parameters $\{\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{A}, \boldsymbol{\mu}\}$ estimated on the *i*-vectors passed through the RBM instead of DAE. This parameter transfer is depicted by arrows on the Figure 1. In more detail, first, we train denoising RBM and apply it to the *training* set of *i*-vectors, second, we estimate the parameters of the whitening transform and PLDA on the RBM outputs, then, we train autoencoder initialized from the RBM (depicted as the dashed rectangle on the Figure 1) and, finally, we apply whitening and PLDA to the *test* set passed though the autoencoder. That is, we apply PLDA to *i*-vectors from the feature space which differs from the space used to train PLDA and whitening. We investigate the effects of such parameter substitution in Section 2.3.1. Whenever we mention the default DAE system, we refer to the configuration described above.

Table 1 presents the results of the described above DAE based speaker verification system evaluation for the NIST SRE 2010 condition 5 extended test. Also in Table 2 we present results for another corpus called "Rus-Telecom". Rus-Telecom is the Russian-language corpus of telephone recordings, collected by the call-centers in Russia. For our experiments we used only male subcorpus. Training set consists of 6508 male speakers and 33678 speech cuts. Evaluation part consists of 235 male speakers and 4210 speech cuts. Evaluation protocol (single-session enrollments) contains 37184 target trials and 111660 impostor trials.

Table 1: Performance comparison of three systems reported on the NIST SRE 2010 test.

System	EER,%	minDCF
Baseline	1.67	0.347
RBM	1.55	0.332
DAE	**1.43**	**0.284**

Table 2: Performance comparison of three systems trained and tested on the Rus-Telecom corpus.

System	EER,%	minDCF
Baseline	1.63	0.644
RBM	1.65	0.632
DAE	**1.43**	**0.557**

Comparing the 1st and 3rd rows in Table 1 and Table 2, we observe that the DAE system considerably outperforms the Baseline system.

2.3. Analysis of the DAE system performance

Here we explore the main causes leading to performance gaps between the Baseline and DAE system showed in Table 1 and consider the impact of each processing unit of the entire system separately. First, in Sections 2.3.1-2.3.2 we try to assess the nonlinear mappings learned by RBM and DAE. Then, in Section 2.3.3 we explore the effect of parameter substitution described in Section 2.2.4.

2.3.1. Assessing denoising transform

We assess the nonlinear mapping learned by DAE independently of the back-end to separate the impacts of DAE and PLDA to the overall error reduction. Intuitively, we aim at minimizing within-speaker variability and maximizing between-speaker variability simultaneously. This aim is formalized by the following class-separability criterion [19]:

$$J = \text{tr}(\boldsymbol{\Sigma}_W^{-1} \boldsymbol{\Sigma}_B) \equiv \text{tr}(\boldsymbol{F}), \tag{4}$$

where $\boldsymbol{\Sigma}_W$ and $\boldsymbol{\Sigma}_B$ are the within-speaker and between-speaker covariance matrices, respectively, estimated from either raw or transformed *i*-vectors. In addition to that, we use cosine scoring to estimate EER and minDCF which can be also used as independent criteria.

Table 3 shows a comparison of four feature spaces in terms of discriminative ability for speaker verification. The first row corresponds to the whitened and length normalized raw *i*-vectors. The next three rows correspond to the transformed *i*-vectors. It should be noted that in the latter cases *i*-vectors were normalized before passing through the nonlinear transform but *no* normalization was applied to their images.

Table 3: Comparison of four feature spaces in terms of speaker verification performance (EER,% minDCF) with cosine back-end and criterion J (higher is better).

System	EER,%	minDCF	J
Baseline	5.34	0.603	501.4
RBM	5.27	0.611	525.6
DAE	3.19	0.427	537.7
AE	5.42	0.583	494.1

As can be seen from Table 3, DAE transform gives the largest value of J and the smallest EER and minDCF. Better performance for cosine scoring can be explained by narrowing the "cone" of speaker's sessions after the nonlinear transform.

In the last row we present the results for the standard autoencoder (AE). In this case we replaced $\boldsymbol{i}(s)$ by $\boldsymbol{i}(s,h)$ in the training objective (1). As expected, the values of J, EER and minDCF for the AE system are comparable to those for the Baseline system. In contrast to DAE, which attempts to minimize within-speaker variability, the objective of AE is to perfectly reconstruct the input and map the *i*-vectors as close as possible to their original locations.

For the more detailed view on class-separability in these feature spaces Figure 3 shows the spectrum of the matrix $\boldsymbol{F}$ for three cases: Baseline, RBM and DAE. As J is independent to a linear transform (*i.e.* given a full-rank $\boldsymbol{C}$, $\text{tr}(\boldsymbol{C}^{-\top} \boldsymbol{\Sigma}_W^{-1} \boldsymbol{C}^{-1} \boldsymbol{C} \boldsymbol{\Sigma}_B \boldsymbol{C}^{\top}) = \text{tr}(\boldsymbol{\Sigma}_W^{-1} \boldsymbol{\Sigma}_B)$), $\boldsymbol{F}^{-1}$ can be viewed as within-class covariance matrix in the space where $\boldsymbol{\Sigma}_B = \mathbf{I}$ (*i.e.* $\boldsymbol{C} = \text{chol}(\boldsymbol{\Sigma}_B^{-1})$).

The gap between the spectrum of the DAE system and spectra of two other systems demonstrates that discriminatively

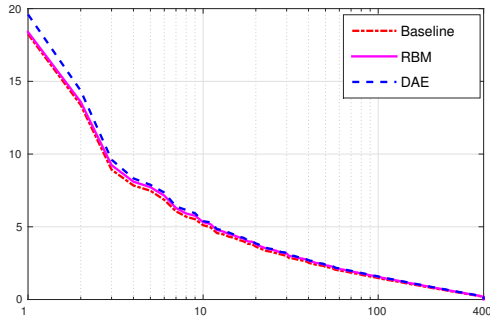

Figure 3: Eigenvalues of the matrix $\boldsymbol{F}$ for the systems corrsponding to the first three rows in Table 3. No normalization was applied to the outputs of RBM and DAE

trained denoising transform increases class-separability of the *i*-vector space.

2.3.2. *Effect of normalization*

Similarly to Table 3, Table 4 shows measurements of discriminative ability of the same three feature spaces. The only difference is that the transformed *i*-vectors were normalized *i.e.* whitened and projected to sphere.

Table 4: Comparison of three feature spaces in terms of speaker verification performance with cosine back-end. RBM and DAE outputs are normalized.

System	EER,%	minDCF
Baseline	5.34	0.603
RBM	4.96	0.565
DAE	4.95	0.558

We can observe that normalization reduces EER for the RBM from 5.36% to 4.96%, but increases it for the DAE from 3.19% to 4.96%. In this case RBM and DAE transforms do not make much difference in terms of cosine similarity. This is consistent with Figure 4 which shows the 10 largest eigenvalues of the matrix $\boldsymbol{F}$ computed for each of three cases.

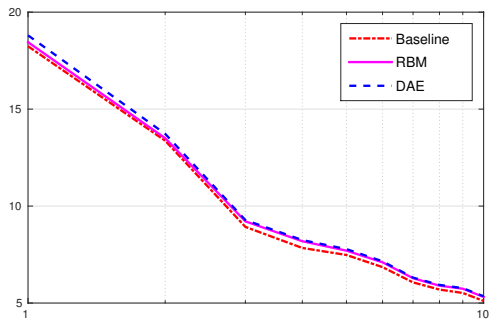

Figure 4: The 10 largest eigenvalues of the matrix $\boldsymbol{F}$ for the same three systems as in Table 4.

As was mentioned in Section 2.2.4, we found that using parameters of the normalization and the back-end classifier trained on the *i*-vectors denoised by RBM instead of DAE leads to the best performance of the DAE system. Table 5 shows considerable reduction of EER and minDCF after replacing $\boldsymbol{A}$ and $\boldsymbol{\mu}$ by the corresponding normalization parameters estimated before discriminative fine-tuning.

Table 5: Comparison of three feature spaces in terms of speaker verification performance with cosine back-end. RBM and DAE outputs are normalized. Whitening and back-end parameters of the DAE system are replaced as described in text.

System	EER,%	minDCF
Baseline	5.34	0.603
RBM	4.96	0.565
DAE	**2.83**	**0.393**

As before, Figure 5 demonstrates the largest eigenvalues of the matrix $\boldsymbol{F}$. We can see that the *i*-vectors denoised by DAE and normalized in the way described above have considerably better discriminative properties.

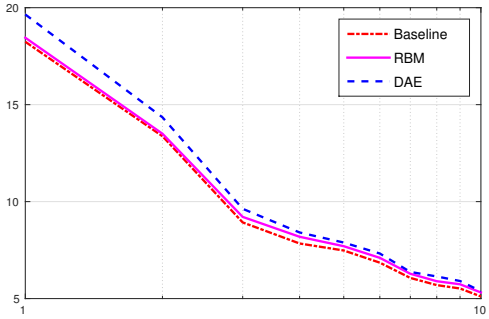

Figure 5: The 10 largest eigenvalues of matrix $\boldsymbol{F}$ for the same three systems as in Table 5.

2.3.3. *Effect of replacing back-end parameters*

In the Section 2.3.2 we found that transferring whitening parameters from the RBM system leads to much better class separability in terms of cosine similarity. Now we explore the effect of joint substitution of $\{\boldsymbol{A}, \boldsymbol{\mu}; \boldsymbol{P}, \boldsymbol{Q}\}$ trained on RBM outputs to the DAE system. Table 6 reports evaluation results for various configurations of the DAE system.

It is important to note that PLDA parameterized by $\{\boldsymbol{P}, \boldsymbol{Q}\}$ (2nd column of Table 6) was trained using whitening parameters estimated on the *same* training data, while the whitening parameters $\{\boldsymbol{A}, \boldsymbol{\mu}\}$ (3rd column in Table 6) for the test data may be different.

Analyzing Table 6, we can make the following conclusions. The performances of the RBM and DAE systems are comparable (2nd and 3rd rows) when all the parameters were estimated on the same corresponding data, which is consistent with Table 4. In this case DAE system (3rd row) yields only moderate improvement over the Baseline.

As we can see in the 6th row in Table 6, the substitution of the whitening matrix $\boldsymbol{A}$ from the RBM system has the largest impact to increasing accuracy. The further substitution of the mean vector $\boldsymbol{\mu}$ (7th row) brings only minor improvement of the DAE system. This can be explained by the fact that for a homogeneous corpus both the global means of DAE and RBM outputs are close zero and almost match each other.

Table 6: Performance comparison for different configurations of the DAE system. (See Section 2.3.3 for details).

System	PLDA: $\{\boldsymbol{P},\boldsymbol{Q}\}$	Whitening: $\boldsymbol{A}/\boldsymbol{\mu}$	EER,%	minDCF
Baseline	raw	raw/raw	1.67	0.347
RBM	RBM	RBM/RBM	1.55	0.332
DAE	DAE	DAE/DAE	1.58	0.336
DAE	DAE	DAE/RBM	1.55	0.338
DAE	RBM	DAE/DAE	1.56	0.33
DAE	DAE	RBM/DAE	1.43	0.291
DAE	DAE	RBM/RBM	1.44	0.287
DAE	RBM	RBM/RBM	**1.43**	**0.284**

Then, using $\{\boldsymbol{P},\boldsymbol{Q}\}$ from the RBM system makes the smallest contribution to filling the performance gap between the 3rd and 8th rows. However, as observed in [1], replacing of the parameters $\{\boldsymbol{P},\boldsymbol{Q}\}$ makes the largest contribution to the accuracy when whitening parameters are kept unchanged (*i.e.* trained on DAE outputs).

Thus, Table 6 demonstrates how much replacing of each parameter in the DAE system by the corresponding one from the RBM system affects the overall accuracy. The question why training system parameters in the different feature space leads to better accuracy is still open.

3. An improved DAE system

In this Section we describe two improvements to our previous system [1].

3.1. Dropout for RBM training

Dropout is the heuristic for training neural networks which can be viewed as a type of regularization to prevent neural network from overfitting [5]. Dropout training has been empirically found to be more efficient than the commonly used L2 regularization [20].

The intuition behind dropout is to randomly drop units from the neural network during training to prevent co-adaptation of units. It can be applied either to the generative training of a RBM or to the discriminative fine-tuning of a neural network [20]. We applied dropout during RBM training. In our case, dropping out the 20% of hidden units was found to be optimal.

Our experiments show (see Table 7) that RBM trained with dropout provides better initialization for DAE before discriminative fine-tuning compared to the training without dropout.

Table 7: Effect of dropout for RBM training. RBM is used to initialize DAE.

System	EER,%	minDCF
DAE	1.43	0.284
DAE+dropout	**1.41**	**0.270**

Unfortunately, applying dropout at the stage of discriminative fine-tuning was not helpful.

3.2. Deep denoising autoencoders

Adopting ideas from the deep learning where autoencoders are usually used as a building block for deep neural networks [6] we make attempts to build such model. We will describe two approaches to build and train deep networks to denoise *i*-vectors.

3.2.1. Stacking RBMs

First, we train a stack of two RBMs according to Figure 6. After training, the first RBM is unfolded and the input vector $\boldsymbol{i}(s,h)$ is mapped to the $\boldsymbol{i}(s)$, which becomes the input for the second RBM. Then the second RBM is unfolded to form the 5-layer denoising neural network, further referred to as deep DAE. Before discriminative fine-tuning it is initialized by two pairs of weight matrices $\{\boldsymbol{W},\boldsymbol{V}^\top\}$ from the corresponding RBMs. It is important to note that all the parameters of the deep DAE are trained *jointly*. We apply whitening and length normalization both to raw *i*-vectors and to the outputs of the network. As described in Section 2.3.3, we transfer parameters from the PLDA trained on the outputs of the second RBM.

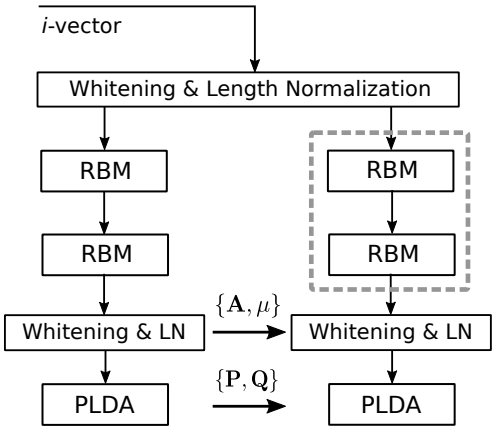

Figure 6: Block diagram for the system based on deep DAE. Each RBM block corresponds to the RBM showed on Figure 2. Dashed rectangle denotes discriminatively trained parts, it surrounds the blocks trained jointly.

Table 8: Comparison of the single DAE and the jointly trained deep DAE initialized from the stack of RBMs. (See Figure 6 for details).

System	EER,%	minDCF
Baseline	1.67	0.347
DAE	1.43	0.284
deep DAE	1.43	0.297

As we can see from Table 8 this strategy does not bring improvement to the performance. In the next Section we describe an alternative approach leading to the better performance.

3.2.2. Stacking DAEs

Here we present another starategy to build a deep model. In contrast to the previously described approach based on the joint training of the 5-layered DAE, we build a stack of two DAEs and train each DAE *separately* applying whitening and length normalization to the mapped vectors. Figure 7 shows detailed pipeline. As before, we transfer parameters of the network, whitening and PLDA. This is denoted by arrows on the diagram.

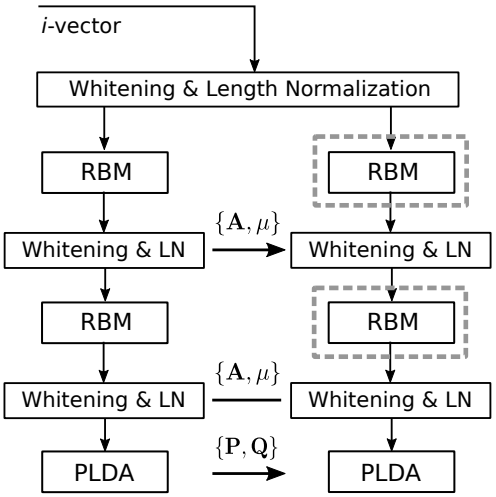

Figure 7: Block diagram for the system based on stack of DAEs. Each RBM block corresponds to the RBM showed on Figure 2. Dashed rectangle denotes discriminative training.

Table 9 compares performance of stacked RBMs and DAEs (denoted by 2-RBM and 2-DAE, respectively) to the variant with the single DAE. This approach allows reducing EER to 1.30% which is the our best result on the NIST SRE 2010 CC5 extended protocol. We did not try to build the stack of three DAEs as we do not expect significant improvement further.

Table 9: Performance of the system based on stacked DAEs trained separately.

System	EER,%	minDCF
Baseline	1.67	0.347
RBM	1.55	0.332
DAE	1.43	0.284
2-RBM	1.58	0.392
2-DAE	**1.30**	**0.282**

4. DAE system in the domain mismatch scenario

Recently, the problem of *domain mismatch* (*i.e.* mismatch of the distribution of inputs between source and target domains, also referred to as domain shift or dataset bias [21]) has attracted careful attention of the speaker recognition community. It was found that domain mismatch leads to considerable performance gap between the in-domain and out-of-domain speaker recognition systems (*i.e.* systems trained on the in-domain and out-of-domain data, respectively) [22]. This motivated to set up the domain adaptation challenge (DAC) which was one of the main topics of interest during the summer workshop held at the Johns Hopkins University (JHU) in 2013 [22]. Domain adaptation aims to effectively exploit scarce in-domain data along with typically more plentiful out-of-domain data whose distributions are different. It assumed that in domain dataset has limited number of labeled samples or class labels are not available due to the high cost of manual annotation. Otherwise a classifier could be accurately trained on the in-domain dataset directly.

In our current and previous studies [1] we have already shown that nonlinear mapping learned by DAE allows to improve speaker verification performance in matched conditions. In this Section we explore the contribution of DAE in the case of domain mismatch.

4.1. Dataset

We followed the DAC setup detailed as follows. The *i*-vector extractor uses 40-dimensional MFCCs (20 base + deltas) with short-term mean and variance normalization. It uses a 2048-component gender-independent UBM with a 600 dimensional gender- independent *i*-vector extractor. As for our previous experiments, the SRE10 telephone data (condition 5 extended task, males) was used for evaluation. This evaluation set consists of 3,465 target and 175,873 non-target trials. For parameter training two datasets are defined. The in-domain SRE set includes telephone calls from 1,115 male speakers and 13,628 speech cuts taken from SRE 04, 05, 06, and 08 collections. The out-of-domain SWB set includes calls from 1,461 male speakers and 15,164 speech cuts taken from Switchboard [11]. More details of this setup can be found at [22]. In our experiments we ignore labels of the in-domain data. We used in-domain SRE set only to estimate the whitening parameters of our systems.

4.2. Back-ends

We use cosine scoring, two-covariance model (referred to as PLDA) and simplified PLDA [8] with 400-dimensional speaker subspace (referred to as SPLDA).

4.3. Results

Table 10 demonstrates performance of three different speaker verification systems for the domain mismatch conditions. The details of training of the RBM and DAE systems can be found in Section 2. We used RBM with one hidden layer trained with dropout to pre-train DAE.

We evaluated these systems under three setups which differ by the datasets used to estimate the whitening transform and PLDA parameters.

The first three rows of Table 10 show evaluation results for the case when both whitening transform and PLDA were estimated on the in-domain (SRE) data. As before, we can see that applying denoising transform reduces EER and minDCF compared to the Baseline system. Next three rows show results for the "opposite" case – no in-domain data is available. In this case DAE brings only little performance improvement. But when in-domain data is used to estimate the whitening parameters we can see a considerable improvement. For the Baseline system EER drops from 6.45% to 4.23%. At the same time DAE system, having EER of 2.63%, achieves 38% of relative error reduction compared to the Baseline. Also in this case cosine based system with DAE transform has comparable performance to the Baseline PLDA.

Thus, from these results we can conclude that using whitening parameters from the target domain along with DAE trained on the out-of-domain set allows to avoid significant performance gap caused by domain mismatch.

Our recipe can be applied in cases where only unlabeled data from a target domain is available. This data can be used to find the mean and the whitening transform for supervised training of the DAE based speaker verification system on a labeled out-of-domain set.

We also evaluated performance of SPLDA on the same setup. Table 11 shows slightly better performance of SPLDA

Table 10: Performance summary of speaker verification systems with PLDA and cosine back-ends. The second column shows which dataset was used to estimate the whitening transform and PLDA parameters.

System	Whitening/ Training	COS		PLDA	
		EER,%	minDCF	EER,%	minDCF
Baseline	SRE/SRE	5.45	0.621	2.18	0.360
RBM		5.47	0.634	2.16	0.348
DAE		**3.67**	**0.467**	**1.67**	**0.307**
Baseline	SWB/SWB	9.13	0.788	6.45	0.660
RBM		8.97	0.778	6.28	0.667
DAE		**8.97**	**0.764**	**6.01**	**0.644**
Baseline	SRE/SWB	5.45	0.621	4.23	0.554
RBM		5.35	0.631	2.97	0.447
DAE		**4.62**	**0.560**	**2.63**	**0.401**

Table 11: Performance of speaker verification systems with SPLDA back-end. The second column shows which dataset was used to estimate whitening transform and to train PLDA parameters.

System	Whitening/ Training	SPLDA	
		EER,%	minDCF
Baseline	SRE/SRE	2.23	0.312
RBM		2.07	0.317
DAE		**1.61**	**0.292**
Baseline	SRE/SWB	4.21	0.531
RBM		2.66	0.410
DAE		**2.36**	**0.400**

while being consistent with Table 10 in terms of relative performance gaps for the different systems.

5. Conclusion

In this paper a comprehensive study of denoising autoencoders in the *i*-vector space is presented. The investigated DAE based speaker verification system firstly introduced in [1] has shown significant improvement for two operating points in comparison to a standard baseline. The improvement has been demonstrated for two independent tasks when training and testing conditions were matched: NIST SRE 2010 and Rus-Telecom.

It is found that the observed performance gain was due to employing a whitening matrix derived from RBM outputs. However we propose to use in practice all parameters $\{\boldsymbol{A}, \boldsymbol{\mu}; \boldsymbol{P}, \boldsymbol{Q}\}$ trained on RBM outputs. We have achieved further improvements for our DAE system with such substitution and with a modified structure and training procedure. The new version of DAE system consists of two stacked denising autoencoders and employs dropout during training.

The key point of this paper is the question of a theoretical basis for the observed gains associated with substitution. Such gains were observed to be reproducible, yet the authors struggled without success to find a convincing theoretical explanation why RBM transform provides better whitening for a test set. We bring this question forward for discussion and we hope to find the answer with the help of the speaker recognition community.

Performance of a system based on denoising autoencoders in domain mismatch conditions has been investigated. It is shown that calculating whitening parameters on in-domain data and using them to train a denoising autoencoder based system on out-of-domain data allows to avoid large performance gap typically arising due to mismatched conditions between training and testing data. In other words one can use unlabeled in-domain data to obtain whitening parameters and use them to preprocess available out-of-domain data before training a DAE based speaker verification system with these data to improve system performance significantly.

All our findings regarding speaker verification systems in matched conditions hold true for mismatched conditions case. The experiments with the DAC datasets support the importance of employing the back-end parameters found using RBM model in a DAE based system and demonstrate that it is necessary to reconcile whitening parameters used during a training phase with those used in a testing phase when mismatched conditions occur.

6. Acknowledgment

This work was financially supported by the Government of the Russian Federation (Grant 074-U01).

7. References

[1] Sergey Novoselov, Timur Pekhovsky, Oleg Kudashev, Valentin Mendelev, and Alexey Prudnikov, "Non-linear PLDA for i-vector speaker verification," in *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, 2015, pp. 214–218.

[2] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[3] G. Hinton, Li Deng, Dong Yu, G.E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, Nov 2012.

[4] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.

[5] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[6] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle, "Greedy layer-wise training of deep networks," in *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, 2006, pp. 153–160.

[7] "The NIST year 2010 speaker recognition evaluation plan," `http://www.itl.nist.gov/iad/mig//tests/sre/2010/NIST_SRE10_evalplan.r6.pdf`, Accessed: 2016-01-29.

[8] Daniel Garcia-Romero and Carol Y Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems.," in *Interspeech*, 2011, pp. 249–252.

[9] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011, number EPFL-CONF-192584.

[10] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," 2014, pp. 1695–1699.

[11] "The lingusitic data consortium (LDC) catalog," `http://catalog.ldc.upenn.edu`, Accessed: 2016-01-29.

[12] Mohammed Senoussaoui, Najim Dehak, Patrick Kenny, Réda Dehak, and Pierre Dumouchel, "First attempt of boltzmann machines for speaker verification," in *Odyssey 2012: The Speaker and Language Recognition Workshop, Singapore, June 25-28, 2012*, 2012, pp. 117–121.

[13] Geoffrey E. Hinton, "A practical guide to training restricted boltzmann machines," in *Neural Networks: Tricks of the Trade - Second Edition*, pp. 599–619. 2012.

[14] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, New York, NY, USA, 2008, ICML '08, pp. 1096–1103, ACM.

[15] Carl E. Rasmussen, "`minimize.m`," `http://www.gatsby.ucl.ac.uk/~edward/code/minimize/`, Accessed: 2016-01-29.

[16] Niko Brümmer and Edward de Villiers, "The speaker partitioning problem," in *Odyssey 2010: The Speaker and Language Recognition Workshop, Brno, Czech Republic, June 28 - July 1, 2010*, 2010, p. 34.

[17] Simon J. D. Prince and James H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*, 2007, pp. 1–8.

[18] Patrick Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Odyssey 2010: The Speaker and Language Recognition Workshop, Brno, Czech Republic, June 28 - July 1, 2010*, 2010, p. 14.

[19] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, 2 edition, 1990.

[20] Nitish Srivastava, "Improving Neural Networks with Dropout," M.S. thesis, University of Toronto, Toronto, Canada, January 2013.

[21] J. Quiñonero Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*, The MIT Press, 2009, (Editors).

[22] "JHU 2013 speaker recognition workshop," `http://www.clsp.jhu.edu/workshops/13-workshop/speaker-and-language-recognition/`, Accessed: 2016-01-29.