Extracting Speaker-Specific Information with a Deep Neural Architecture

Ahmad Salman and Ke Chen, *Senior Member, IEEE*

Abstract—It is well known that interference among various information components conveyed in speech, such as linguistic and speaker-specific information (SSI), hinders either speaker or speech recognition system from yielding a better performance. In this paper, we present a deep neural architecture (DNA) especially for extracting SSI via learning from Mel-frequency cepstral coefficients (MFCCs), a speech representation commonly used in speech information processing, to facilitate speaker recognition. For learning speaker-specific characteristics, we propose a novel multi-objective loss function towards intrinsic SSI extraction along with a minimal information loss based on first- and second-order speaker-dependent statistics in a high-level yet abstract representation space. For training our DNA, we adopt a two-stage hybrid learning strategy, i.e., unsupervised greedy layer-wise learning to initialize parameters and supervised discriminative learning for an optimal solution in terms of our proposed loss function. By using several Linguistic Data Consortium (LDC) benchmark and multi-lingual speech corpora of different variabilities with cross-corpora and cross-language experimental protocols, we investigate the importance of both architecture depth and training data in our DNA learning for SSI extraction. Also we demonstrate extracted SSI by vowel distribution visualization. In comparison to start-of-the-art techniques, experimental results suggest that, incorporated into a simple speaker modeling technique, our generic speaker-specific representations are robust against various mismatches ranging from channels to spoken languages and hence lead to the favorable performance in various speaker recognition tasks.

Index Terms—Speaker-specific information extraction, deep neural architecture, multi-objective loss functions, data regularization, hybrid learning strategy, speaker recognition, speech information component analysis.

1 Introduction

Automatic *speaker recognition* (SR) generally involves three aspects: speaker-specific feature extraction, *speaker modeling* (SM) and *decision making* (DM) [1], [2]. Recently, attention has been mainly devoted to SM and DM in SR studies and substantial progresses have been made in the two aspects [1]-[3]. For SM, generative models such as *Gaussian mixture model* (GMM) are among the most successful pragmatic approaches by approximating speaker-specific distribution from speech [4]-[6]. On the other hand, discriminative models, e.g., kernel-based learning, have been presented to enhance GMMs to deal with inter- and intra-speaker variabilities in high-level parametric space [7], [8]. Also several compensation techniques [6]-[9], e.g., decomposition of speaker-related and environmental variabilities during SM, have been proposed to tackle the notorious mismatch problem in SR, which furthermore reinforces stat-of-the-art SM techniques. A number of DM techniques [3] are also developed for building up practical SR systems. While all the aforementioned SM and DM techniques significantly improves SR performance, speaker-specific feature extraction, a core aspect in SR, seems to be overlooked, to a great extent, due to the well-known challenge that *linguistic information* (LI) and *speaker-specific information* (SSI) conveyed in speech are fundamentally intermingled and difficult to separate [10]-[14]. As a result, almost all existing speech representations [15], [16], e.g., various spectral representations, carry all kinds of mixing information as a whole and are used in both SR and speech recognition. Mutual interference among various speech information components becomes one of main hurdles to hinder either SR or speech recognition system from yielding a better performance [17], [18].

In previous work, several attempts have been made to explicitly or implicitly extract acoustic features sensitive to the speaker variation [19]. The explicit feature extraction includes natural acoustic features like pitch, intensity, vocal tract filter modeling, glottal flow derivatives, source onset timings and so on [20], [21]. Such features are influenced by a speaker's vocal apparatus. However, such features still convey mixed information components despite an emphasis on SSI. In addition, higher-level prosodic features are also related to emotional stats and speaking style of the speaker [22], [23]. As non-spectral features, those source-related and prosodic features are generally computationally expensive and quite sensitive to the intra-speaker and environmental variabilities. For implicit feature extraction, statistical data analysis techniques, e.g., *principal component analysis* (PCA) and *independent component analysis* (ICA), have been attempted to segregate the SSI from LI [24]. With the assumption that SSI is uncorrelated with other non-SSI and statistically independent among different speakers, PCA or ICA performs unsupervised learning with general objectives regardless of task-specific information. Hence, it is unjustifiable that features extracted by PCA or ICA always encode SSI, and the influence of predom-

Ahmad Salman and Ke Chen are with School of Computer Science, The University of Manchester, Manchester M13 9PL, U.K. Email: {salmanaa, chen}@cs.manchester.ac.uk

inant LI along with noise and channel variabilities often exacerbate the problem. Hence, implicit feature extraction methods often overfit to an observed data set but fail to extract intrinsic SSI. In addition, discriminative learning has also been applied to SSI extraction to establish speaker-specific mapping for individual speakers, which leads to improved SR performance [25], [26]. Despite the limited success in previous work, intrinsic SSI extraction for robust SR remains unresolved in general [17], [18].

As an emerging machine learning methodology, deep learning (DL) employs a deep architecture (DA) composed of multiple levels of non-linear operations to tackle complex AI problems by learning the desired high-level abstraction of input data in a hierarchical way [27]-[29]. DL leads to hierarchical yet distributed representations that re-distribute information conveyed in input data to facilitate complex problem solving and robust DM [29]. DL has been successfully applied to a number of difficult AI tasks ranging from computer vision to acoustic modeling [27]-[33].

Motivated by enormous success of deep learning, we recently proposed a *deep neural architecture* (DNA) to learn speaker-specific characteristics from MFCCs [34]. While we demonstrated that the deep learning leads to a speaker-specific representation, there are still several unsolved problems as discussed in [34]. First, the loss function defined at a single short frame level encounters a difficulty in capturing intrinsic SSI for better generalization. Next, our work was limited by only few speech corpora available at that time. It is observed from our previous work [34] that representations by our DNA fail to yield a satisfactory performance in the presence of severe mismatches. Furthermore, roles of architecture depth and training data is unclear in SSI extraction. Finally, evaluation in our earlier work [34] was not thorough again due to limited speech corpora available.

In this paper, we further develop our DNA to tackle all the problems mentioned above towards extracting intrinsic SSI for robust SR and thoroughly evaluate performance of our improved DNA under different conditions. The contributions of this paper are summarized as follows. First, we propose a novel contrastive loss function based on statistics of long speech segments rather than short individual frames given the fact that statistics of a long speech segment is likely to convey SSI [1], [2] and also facilitate SM used in SR tasks. Using the new loss function, we derive an alternative update rule and empirically show that it performs significantly better than its counterpart [34] in SSI extraction. Second, we investigate the importance of architecture depth in our DNA learning and demonstrate the extracted SSI by vowel distribution visualization. Third, by using elaborately designed experimental protocols, we empirically show that the use of speech corpora covering considerable variabilities in our DNA learning yields robust representations against severe mismatches including text, language, ageing, channel, and environments. Finally, by comparative studies, we demonstrate

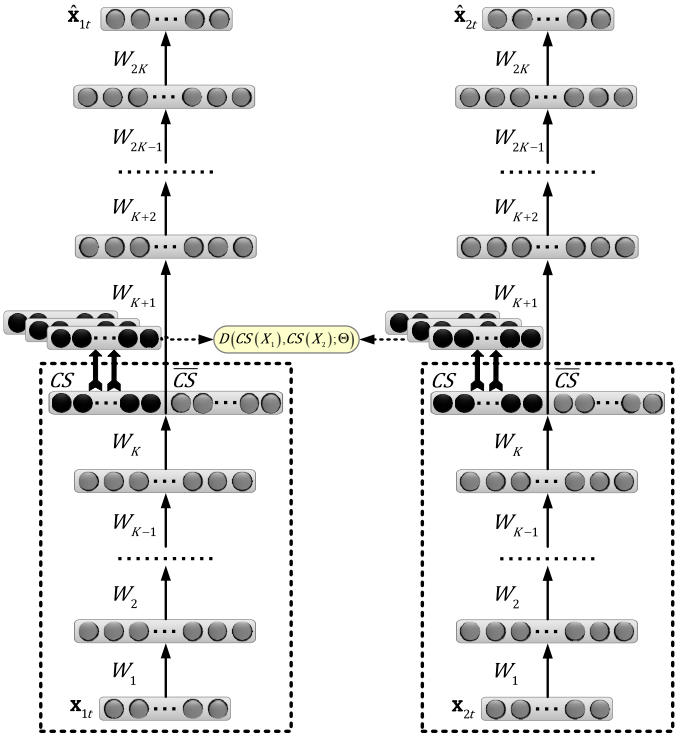

Fig. 1. Deep neural architecture for extracting speaker-specific information.

that resultant representations incorporated into simple speaker modeling techniques lead to the state-of-the-art performance in speaker verification and segmentation tasks. To the best of our knowledge, the work presented in this paper is the first attempt towards extracting intrinsic SSI with machine learning for a generic speaker-specific representation .

The rest of the paper is organized as follows. Sect. II presents our DNA and its learning algorithm. Sect. III describes our experimental methodology and reports experiments results related to our DNA learning. Sect. IV presents comparative studies in two typical SR tasks. Sect. V discusses relevant issues and relates previous work to ours, and the last section draws conclusions.

2 Model Description

In this section, we first describe our deep neural architecture (DNA) designed especially for extracting SSI by learning the statistical compatibility among speakers. Then we present a two-stage learning algorithm by applying the hybrid learning strategy [28], [29] to our proposed loss function to train our DNA.

2.1 Architecture

As illustrated in Fig. 1, our DNA consists of two subnets, and each subnet is a fully connected multi-layered perceptron of *2K+1* layers, i.e., an input layer, *2K-1* hidden layers and a visible layer at the top. If we stipulate that layer *0* is input layer, there are the same number of neurons in layers k and *2K-k* for $k = 0, 1, \cdots, K$. In

particular, the *K*th hidden layer is used as *code layer*, and neurons in this layer are further divided into two subsets denoted by $\mathcal{CS}$ and $\overline{\mathcal{CS}}$, respectively, as depicted in Fig. 1. Those neurons in the subset $\mathcal{CS}$, colored in black in Fig. 1, are used to encode SSI while all the neurons in $\overline{\mathcal{CS}}$ are expected to accommodate non-speaker related information. The input to each subnet is an MFCC representation of a frame after a short-term analysis that a speech segment is divided into a number of frames and the MFCC representation is achieved for each frame. As depicted in Fig. 1, $\boldsymbol{x}_{it}$ is the MFCC feature vector of frame t in X_i, input to subnet i (*i*=1,2), where $X_i = \{\boldsymbol{x}_{it}\}_{t=1}^{T_B}$ collectively denotes MFCC feature vectors for a speech segment of T_B frames.

During learning, two identical subsets are coupled at their coding layers via neurons in $\mathcal{CS}$ with an incompatibility measure defined on two speech segments of equal length, X_1 and X_2, input to two subnets, which will be presented in 2.2. Thus, our DNA can be regarded as a variant of Siamese neural architecture [35], a regularized version as elucidated in 2.2. After learning, we achieve two identical subnets and hence can use either of them to produce a new representation for a speech frame. For input $\boldsymbol{x}$ to a subnet, only the bottom K layers of the subnet are used and the output of neurons in $\mathcal{CS}$ at the code layer or layer K, denoted by $\mathcal{CS}(\boldsymbol{x})$, is its new representation, as illustrated by the dash box in Fig. 1.

2.2 Loss Function

Let $\mathcal{CS}(\boldsymbol{x}_{it})$ be the output of all neurons in $\mathcal{CS}$ of subnet i (*i*=1,2) for input $\boldsymbol{x}_{it} \in X_i$ and $\mathcal{CS}(X_i) = \{\mathcal{CS}(\boldsymbol{x}_{it})\}_{t=1}^{T_B}$, which pools output of neurons in $\mathcal{CS}$ for T_B frames in X_i, as illustrated in Figure 1. As statistics of speech signals is more likely to capture SSI [5], we define the incompatibility measure based on the 1st- and 2nd-order statistics of a new representation to be learned as

$$D[\mathcal{CS}(X_1), \mathcal{CS}(X_2); \Theta] = ||\boldsymbol{\mu}^{(1)} - \boldsymbol{\mu}^{(2)}||_2^2 + ||\Sigma^{(1)} - \Sigma^{(2)}||_F^2, \quad (1)$$

where

$$\boldsymbol{\mu}^{(i)} = \frac{1}{T_B}\sum_{t=1}^{T_B} \mathcal{CS}(\boldsymbol{x}_{it}),$$

$$\Sigma^{(i)} = \frac{1}{T_B - 1}\sum_{t=1}^{T_B} [\mathcal{CS}(\boldsymbol{x}_{it}) - \boldsymbol{\mu}^{(i)}][\mathcal{CS}(\boldsymbol{x}_{it}) - \boldsymbol{\mu}^{(i)}]^T,\ i = 1, 2.$$

In Eq. (1), $||\cdot||_2$ and $||\cdot||_F$ are the $\mathcal{L}_2$ norm and the Frobenius norm, respectively. Θ is a collective notation of all connection weights and biases in the DNA. Intuitively, two speech segments belonging to different speakers lead to different statistics and hence their incompatibility score measured by (1) should be large after learning. Otherwise their score is expected to be small.

For a corpus of multiple speakers, we can construct a training set so that an example be in the form: $(X_1, X_2; \mathcal{I})$ where $\mathcal{I}$ is the label defined as $\mathcal{I} = 1$ if two speech segments, X_1 and X_2, are spoken by the same speaker or $\mathcal{I} = 0$ otherwise. Using such training examples, we apply the energy-based model principle [36] to define a loss function as

$$\begin{aligned} L(X_1, X_2; \Theta) &= \alpha[L_R(X_1; \Theta) + L_R(X_2; \Theta)] \\ &\quad + (1 - \alpha) L_D(X_1, X_2; \Theta), \end{aligned} \quad (2)$$

where

$$L_R(X_i; \Theta) = \frac{1}{T_B}\sum_{t=1}^{T_B} ||\boldsymbol{x}_{it} - \hat{\boldsymbol{x}}_{it}||_2^2 \ (i = 1, 2), \quad (3a)$$

$$L_D(X_1, X_2; \Theta) = \mathcal{I}D + (1 - \mathcal{I})(e^{-\frac{D_m}{\lambda_m}} + e^{-\frac{D_S}{\lambda_S}}). \quad (3b)$$

Here $D_m = ||\boldsymbol{\mu}^{(1)} - \boldsymbol{\mu}^{(2)}||_2^2$ and $D_S = ||\Sigma^{(1)} - \Sigma^{(2)}||_F^2$. λ_m and λ_S are the tolerance bounds of incompatibility scores in terms of D_m and D_S, which can be estimated from a training set. In $L_D(X_1, X_2; \Theta)$, we drop explicit parameters of $D[\mathcal{CS}(X_1), \mathcal{CS}(X_2); \Theta]$ to simplify presentation.

Eq. (2) defines a multi-objective loss function where α $(0 < \alpha < 1)$ is a parameter used to trade-off between two objectives $L_R(X_i; \Theta)$ and $L_D(X_1, X_2; \Theta)$. The motivation for two objectives are as follows. By nature, both SSI and non-speaker related information components are entangled over speech [16], [34]. When we tend to extract SSI, the interference of non-speaker related information is inevitable and appears in various forms. $L_D(X_1, X_2; \Theta)$ measures errors responsible for wrong speaker-specific statistics on a representation learned by a Siamese DA in different situations. However, using $L_D(X_1, X_2; \Theta)$ only to train a Siamese DA cannot cope with enormous variations of non-speaker related information, e.g., LI (a predominant information component in speech), which often leads to overfitting to a training corpus according to our observations [34]. As a result, we use $L_R(X_i; \Theta)$ to measure reconstruction errors to monitor information loss during SSI extraction. By minimizing reconstruction errors in two subnets, the code layer leads to a speaker-specific representation with the output of neurons in $\mathcal{CS}$ while the remaining neurons are used to regularize various interference by capturing some invariant properties underlying them for good generalization.

In summary, we anticipate that minimizing the multi-objective loss function defined in Eq. (2) will enable our DNA to extract SSI by encoding it through a generic speaker-specific representation insensitive to various mismatches as well as, more importantly, applicable to other speech corpora that has never been seen in training the DNA, which will be verified by our experiments reported later on.

2.3 Learning Algorithm

In this section, we apply the two-phase deep learning strategy [28], [29] to derive our learning algorithm, i.e., pre-training for initializing subnets and discriminative learning for learning a speaker-specific representation.

We first present the notation system used in our algorithm. Let $h_{kj}(\boldsymbol{x}_{it})$ denote the output of the *j*th neuron

in layer k for k=0,1,$\cdots$,K,$\cdots$,2K. $\boldsymbol{h}_k(\boldsymbol{x}_{it}) = \big(h_{kj}(\boldsymbol{x}_{it})\big)_{j=1}^{|\boldsymbol{h}_k|}$ is a collective notation of the output of all neurons in layer k of subnet i (i=1,2) where $|\boldsymbol{h}_k|$ is the number of neurons in layer k. By this notation, k=0 refers to the input layer with $\boldsymbol{h}_0(\boldsymbol{x}_{it}) = \boldsymbol{x}_{it}$, and k=2K refers to the top layer producing the reconstruction $\hat{\boldsymbol{x}}_{it}$. In the coding layer, i.e., layer K, $\mathcal{CS}(\boldsymbol{x}_{it}) = \big(h_{Kj}(\boldsymbol{x}_{it})\big)_{j=1}^{|\mathcal{CS}|}$ is a simplified notation for output of neurons in $\mathcal{CS}$, corresponding to a speaker-specific representation of $\boldsymbol{x}_{it}$ after learning. Let $W_k^{(i)}$ and $\boldsymbol{b}_k^{(i)}$ denote the connection weight matrix between layers k-1 and k and the bias vector of layer k in subnet i (i=1,2), respectively, for k=1,$\cdots$,2K. Then output of layer k is $\boldsymbol{h}_k(\boldsymbol{x}_{it}) = \sigma[\boldsymbol{u}_k(\boldsymbol{x}_{it})]$ for k=1,$\cdots$,2K-1, where $\boldsymbol{u}_k(\boldsymbol{x}_{it}) = W_k^{(i)}\boldsymbol{h}_{k-1}(\boldsymbol{x}_{it}) + \boldsymbol{b}_k^{(i)}$ and $\sigma(\boldsymbol{z}) = \big((1+e^{-z_j})^{-1}\big)_{j=1}^{|\boldsymbol{z}|}$. Note that we use the linear transfer function in the top layer, i.e., layer 2K, to reconstruct the original input.

2.3.1 Pre-training

For pre-training, we employ the denoising autoencoder [37] as a building block to initialize biases and connection weight matrices of a subnet. A denoising autoencoder is a three-layered perceptron where the input, $\tilde{\boldsymbol{x}}$, is a distorted version of the target output, $\boldsymbol{x}$. For a training example, $(\tilde{\boldsymbol{x}}, \boldsymbol{x})$, the output of the autoencoder is a restored version, $\hat{\boldsymbol{x}}$. Since MFCCs fed to the first hidden layer and its intermediate representation input to all other hidden layers are of continuous value, we always distort input, $\boldsymbol{x}$, by adding Gaussian noise to form a distorted version, $\tilde{\boldsymbol{x}}$. The restoration learning is done by minimizing the MSE loss between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ with respect to the weight matrix and biases. We apply the stochastic back-propagation (SBP) algorithm to train denoising autoencoders, which is detailed in the appendix of [34], and the greedy layer-wise learning procedure [28], [29] leads to initial weight matrices for the first K hidden layers, as depicted in a dash box in Fig. 1, i.e., $W_1, \cdots, W_K$ of a subnet. Then, we set $W_{K+k} = W_{K-k+1}^T$ for k=1,$\cdots$,K to initialize $W_{K+1}, \cdots, W_{2K}$ of the subnet. Finally, the second subnet is created by simply duplicating the pre-trained one.

2.3.2 Discriminative Learning

For discriminative learning, we minimize the loss function in Eq. (2) based on pre-trained subnets for SSI extraction. Given our loss function is defined on statistics of T_B frames in a speech segment, we cannot update parameters until we have T_B output of neurons in $\mathcal{CS}$ at the code layer. Fortunately, the SBP algorithm perfectly meets our requirement; In the SBP algorithm, we always set the batch size to the number of frames in a speech segment. To simplify the presentation, we shall drop explicit parameters in our derivation whenever doing so causes no ambiguities.

In terms of the reconstruction loss, $L_R(X_i;\Theta)$, we have the following gradients. For layer k = 2K,

$$\frac{\partial L_R}{\partial \boldsymbol{u}_{2K}(\boldsymbol{x}_{it})} = 2(\hat{\boldsymbol{x}}_{it} - \boldsymbol{x}_{it}), \quad i=1,2. \tag{4}$$

For all hidden layers, k=2K-1,$\cdots$,1, applying the chain rule and (4) leads to

$$\frac{\partial L_R}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{it})} = \left(\frac{\partial L_R}{\partial h_{kj}(\boldsymbol{x}_{it})} h_{kj}(\boldsymbol{x}_{it})[1-h_{kj}(\boldsymbol{x}_{it})]\right)_{j=1}^{|\boldsymbol{h}_k|}, \tag{5a}$$

$$\frac{\partial L_R}{\partial \boldsymbol{h}_k(\boldsymbol{x}_{it})} = \big[W_{k+1}^{(i)}\big]^T \frac{\partial L_R}{\partial \boldsymbol{u}_{k+1}(\boldsymbol{x}_{it})}. \tag{5b}$$

As the contrastive loss, $L_D(X_1, X_2; \Theta)$, defined on neurons in $\mathcal{CS}$ at code layers of two subnets, its gradients are determined only by parameters related to K hidden layers in two subnets, as depicted by dash boxes in Fig. 1. For layer k=K and subnet i=1, 2, we obtain

$$\begin{aligned}\frac{\partial L_D}{\partial \boldsymbol{u}_K(\boldsymbol{x}_{it})} &= \Big(\big([\mathcal{I} - \lambda_m^{-1}(1-\mathcal{I})e^{-\frac{D_m}{\lambda_m}}]\psi_j(\boldsymbol{x}_{it})\big)_{j=1}^{|\mathcal{CS}|}, \vec{\boldsymbol{0}}\Big) + \\ &\quad \Big(\big([\mathcal{I} - \lambda_S^{-1}(1-\mathcal{I})e^{-\frac{D_S}{\lambda_S}}]\xi_j(\boldsymbol{x}_{it})\big)_{j=1}^{|\mathcal{CS}|}, \vec{\boldsymbol{0}}\Big). \end{aligned}\tag{6}$$

Here, $\psi_j(\boldsymbol{x}_{it}) = p_j^{(i)}\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big[1-\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big]$ and $\xi_j(\boldsymbol{x}_{it}) = q_j(\boldsymbol{x}_{it})\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big[1-\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big]$, where $\boldsymbol{p}^{(i)} = \frac{2}{T_B}\text{sign}(1.5-i)(\boldsymbol{\mu}^{(1)}-\boldsymbol{\mu}^{(2)})$, $\boldsymbol{q}(\boldsymbol{x}_{it}) = \frac{4}{T_B-1}\text{sign}(1.5-i)(\Sigma^{(1)}-\Sigma^{(2)})[\mathcal{CS}(\boldsymbol{x}_{it})-\boldsymbol{\mu}^{(i)}]$ and $\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j$ is output of the jth neuron in $\mathcal{CS}$ for input $\boldsymbol{x}_{it}$. In Eq. (6), $\vec{\boldsymbol{0}}$ is a zero vector of $|\mathbf{h}_K|-|CS|$ elements corresponding to the gradients of the loss function L_D, defined in Eq. (3b), with respect to potentials of all the neurons in $\overline{\mathcal{CS}}$ (c.f. Fig. 1) i.e., $\vec{\boldsymbol{0}} = (0)_{j=|CS|+1}^{|\mathbf{h}_K|}$. The derivation of Eq. (6) appears in the appendix.

For layers k=K-1, $\cdots$,1, we have

$$\frac{\partial L_D}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{it})} = \left(\frac{\partial L_D}{\partial h_{kj}(\boldsymbol{x}_{it})} h_{kj}(\boldsymbol{x}_{it})[1-h_{kj}(\boldsymbol{x}_{it})]\right)_{j=1}^{|\boldsymbol{h}_k|}, \tag{7a}$$

$$\frac{\partial L_D}{\partial \boldsymbol{h}_k(\boldsymbol{x}_{it})} = \big[W_{k+1}^{(i)}\big]^T \frac{\partial L_R}{\partial \boldsymbol{u}_{k+1}(\boldsymbol{x}_{it})}. \tag{7b}$$

Given a training example, $\big(\{\boldsymbol{x}_{1t}\}_{t=1}^{T_B}, \{\boldsymbol{x}_{2t}\}_{t=1}^{T_B}; \mathcal{I}\big)$, we use gradients achieved from Eqs. (4)-(7) to update all the parameters in the DNA. For layers k=K+1, $\cdots$, 2K, their parameters are updated by

$$W_k^{(i)} \leftarrow W_k^{(i)} - \frac{\epsilon\alpha}{T_B}\sum_{t=1}^{T_B}\sum_{r=1}^{2}\frac{\partial L_R}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{rt})}[\boldsymbol{h}_{k-1}(\boldsymbol{x}_{rt})]^T, \tag{8a}$$

$$\boldsymbol{b}_k^{(i)} \leftarrow \boldsymbol{b}_k^{(i)} - \frac{\epsilon\alpha}{T_B}\sum_{t=1}^{T_B}\sum_{r=1}^{2}\frac{\partial L_R}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{rt})}. \tag{8b}$$

For layers k=1, $\cdots$, K, their weight matrices and biases are updated with

$$\begin{aligned} W_k^{(i)} \leftarrow\ & W_k^{(i)} - \frac{\epsilon}{T_B}\sum_{t=1}^{T_B}\sum_{r=1}^{2}\Big(\alpha\frac{\partial L_R}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{rt})} + \\ & (1-\alpha)\frac{\partial L_D}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{rt})}\Big)[\boldsymbol{h}_{k-1}(\boldsymbol{x}_{rt})]^T, \end{aligned}\tag{9a}$$

$$\boldsymbol{b}_k^{(i)} \leftarrow \boldsymbol{b}_k^{(i)} - \frac{\epsilon}{T_B}\sum_{t=1}^{T_B}\sum_{r=1}^{2}\Big(\alpha\frac{\partial L_R}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{rt})}+(1-\alpha)\frac{\partial L_D}{\partial \boldsymbol{u}_k(\boldsymbol{x}_{rt})}\Big). \quad (9b)$$

In Eqs. (8) and (9), ϵ is a learning rate. Here we emphasize that using sum of gradients with respect to parameters of two subnets in update rules guarantees that two subsets are always kept identical during learning.

3 EXPERIMENTS

In this section, we present several experiments designed to investigate critical issues and properties in learning speaker-specific characteristics. We first describe our experimental methodology and then present two experiments to examine roles of architecture depth and training data in our DNA learning. Finally, we visualize vowel distributions to illustrate the SSI extracted by our DNA learning.

3.1 Experimental Methodology

Now we describe the general setting for DNA learning and then other enabling techniques for our experiments.

3.1.1 Experimental Settings for DNA Learning

For learning speaker-specific characteristics, we employ the English corpus, TIMIT, and all of its variants to train our DNA. TIMIT is one the the most important benchmark corpora for both speaker and speech recognition as it has a large speaker population and utterances of rich LI contains all phonemes of American English [38]. As summarized in TABLE 1, TIMIT and its three variants collected in LDC [38], CTIMIT, HTIMIT and NTIMIT, cover most of possible variabilities or mismatches appearing in SR. To simulate more channel effects, we distort all the utterances in TIMIT by the additive white noise channel with SNR of 10dB and the Rayleigh fading channel with 5 Hz Doppler shift [39], respectively, which generates a simulated noisy corpus of two data sets dubbed *SNTIMIT* listed in TABLE 1. In our experiments presented in this section, we use another LDC English benchmark corpus, KING, and two non-English corpora, CHN and RUS, for test and all of which were collected to evaluate SR systems. As summarized in TABLE 1, KING contains wide-band and narrow-band sets, WKING and NKING, involving all possible variabilities, while CHN [40] and RUS [41] are two corpora in Chinese and Russian, respectively. Here, we emphasize that such a setting allows us to use the cross-corpora and the cross-language protocols, reflecting various mismatches, for evaluating the generalization capability of our generic representations.

In our experiments, we adopt the MFCCs, which encodes many favorable properties of auditory system [16], to be a raw acoustic representation of speech. MFCCs have been widely used in various speech information processing tasks and lead to state-of-the-art performance in both speaker and speech recognition [1], [2], [16]. The same procedure as used in [4]-[6] is applied to all corpora to extract MFCCs as follows: (i) removing silent parts in speech signals with an energy-based method, (ii) pre-emphasis with the filter $H(z)=1–0.95z^{-1}$, (iii) Hamming windowing speech by a frame size of 20 ms with a frame shift of 10 ms, (iv) applying 24 Mel-scale triangular filters to calculate magnitude spectrum, and (v) extracting 20-order MFCCs by excluding the coefficient of order zero.

TABLE 1
Information on corpora used in our experiments.

Corpus	# Speaker	Bandwidth	Variability
TIMIT	630	0-8 kHz	speaker
CTIMIT	462	0.3-3.3 kHz	cellular, speaker
HTIMIT	384	0-4 kHz	handset, speaker
NTIMIT	630	0.3-3.3 kHz	channel, speaker
SNTIMIT	630	0.3-3.3 kHz	cellular, channel, speaker
NKING	51	0.3-3.3 kHz	ageing, channel, handset, speaker
WKING	51	0-4 kHz	ageing, environment, speaker
RT03	N/A	0-4 kHz	speaker
SRE03	356	0.3-3.3 kHz	cellular, speaker
CHN	59	0-8 kHz	ageing, language, speaker
RUS	50	0-4 kHz	language, speaker

For the DNA learning, we divide speakers in a corpus into two groups with a consideration of gender balance and their utterances are used for training and validation. For training, we randomly choose 600 speakers from TIMIT, NTIMT and SNTIMIT, 430 and 350 speakers from CTIMIT and HTIMIT, respectively. The remaining speakers in all five corpora are reserved for validation. For an utterance in the training set, we randomly partition it into speech segments of a length T_B ($1\ \text{sec} \le T_B \le 2\ \text{sec}$) and then exhaustively combine them to form training examples as described in Sect. 2.2. By cross validation, we conduct model selection from a large number of candidate DNAs with $2K+1$ ($1 \le K \le 5$) layers (c.f. Fig. 1) and 50-1000 neurons in a hidden layer. Parameters used in our learning are as follows: Gaussian noise of $N(0, 0.1\sigma)$ used in the denoising autoencoder [37], α=0.2, λ_m=100 and λ_S=2.5 in the loss function defined in Eq. (2) and (3), and learning rates ϵ=0.01 and 0.001 for pre-training and discriminative learning. To avoid overfitting, the number of epochs for discriminative learning is determined by an early stopping criterion.

3.1.2 Speaker Modeling, Distance and Comparison

For any SR tasks, speaker modeling (SM) is inevitable. In our experiments, we use the 1st- and 2nd-order statistics of a speech segment of $|X|$ frames, $X=\{\boldsymbol{x}_t\}_{t=1}^{|X|}$ where $\boldsymbol{x}_t$ is a feature vector of frame t, $\mathcal{SM}=\{\boldsymbol{\mu}, \Sigma\}$, for SM. Furthermore, all utterances belonging to an individual are assumed to follow the normal distribution, $N(\boldsymbol{x}_t|\boldsymbol{\mu}, \Sigma)$, which is well-known as the *mono-Gaussian speaker model*, a simple yet popular SM method for more than two decades [1], [42].

To measure similarity between two speaker models ($\mathcal{SM}$s) in our experiments, we employ the symmetric negative log-likelihood [42] and a variant of the KL divergence [1], [34] for mono-Gaussian SM. Suppose that $\mathcal{SM}_1$ and $\mathcal{SM}_2$ are established with two speech seg-

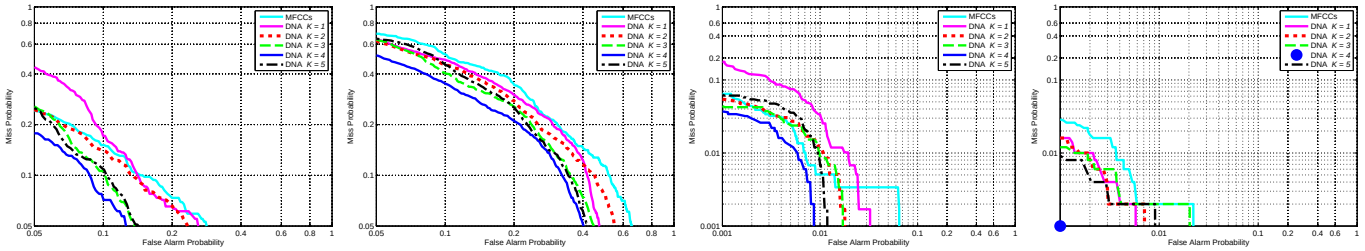

Fig. 2. SC performance (DET curves) of representations generated by our DNA of $2K-1$ hidden layers ($K=1,\cdots,5$) vs. MFCCs on different test corpora. (a) WKING. (b) NKING. (c) CHN. (d) RUS.

ments, $X_1=\{\boldsymbol{x}_{t,1}\}_{t=1}^{|X_1|}$ and $X_2=\{\boldsymbol{x}_{t,2}\}_{t=1}^{|X_2|}$, respectively. The distance metric based on the symmetric negative likelihood [42] is defined as

$$d(\mathcal{SM}_1,\mathcal{SM}_2)=\frac{|X_1|\bar{L}(X_1,X_2)+|X_2|\bar{L}(X_2,X_1)}{|X_1|+|X_2|}, \quad (10)$$

where $\bar{L}(Y,Z)=-\frac{1}{|Z|}\sum_{t=1}^{|Z|}\log N(\boldsymbol{z}_t|\boldsymbol{\mu}_Y,\Sigma_Y)$ and $\boldsymbol{\mu}_Y$ and Σ_Y are mean vector and covariance matrix estimated on a speech segment Y. Although this distance metric works for mono-Gaussian $\mathcal{SM}$s in general, we observe that it does not perform well for $\mathcal{SM}$s established based on very short speech segments irrespective of representations. To tackle this problem in our previous work [34], we defined an alternative distance metric as

$$d(\mathcal{SM}_1,\mathcal{SM}_2)=\mathrm{tr}[(\Sigma_1^{-1}+\Sigma_2^{-1})(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T], \quad (11)$$

where $\boldsymbol{\mu}_i$ and Σ_i $(i=1,2)$ are mean vector and covariance matrix of two $\mathcal{SM}$s built up based on speech segments, X_1 and X_2. The distance metric in (11) was derived from the KL divergence of two normal distributions [1] by dropping the term concerning only covariance matrices that often appears unstable for very short segments [34].

Speaker comparison (SC) is an essential process involved in any SR tasks by comparing two speaker models to collect evidence for DM, which provides a direct way to evaluate representations/speaker modeling [1], [2]. In our experiments presented in this section, we evaluate the SC performance on different representations on the same condition: given a representation, a mono-Gaussian $\mathcal{SM}$ is always built up with a speech segment of a fixed length and the similarity between two $\mathcal{SM}$s is measured by the distance metric in (10). As result, we first divide all the utterances in a test corpus into speech segments of 5 sec to build up $\mathcal{SM}$s as a short utterance poses a greater challenge to SR [1], [2], [5]. Then we exhaustively combine any two $\mathcal{SM}$s to generate $\mathcal{SM}$ pairs. If both $\mathcal{SM}$s in a pair belong to the same speaker, they form a *genuine pair*, and *imposter pair* otherwise. For performance evaluation, we use the *detection error tradeoff* (DET) measure [43] to show all possible errors made in DM during SC to determine whether two $\mathcal{SM}$s are the genuine pair or not, where the area of operating region enclosed by a DET curve and two error axes is generally regarded as the best performance index for SR tasks.

3.2 Validating the Role of Architecture Depth

Model selection is essential for a machine learning system. Due to the limited space here, more results in model selection can be found from [44] and relevant issues will be discussed later on. In this section, we focus on experiments regarding the role of architecture depth in learning speaker-specific representations.

As described in Sect. 3.1.1, model selection in our DNA learning involves many candidate models of a different number of hidden layers where a hidden layer may have a various number of hidden neurons. In our experiment, we use the same training data and learning algorithms described in 3.1.1 to train DNAs of $2K-1$ hidden layers (c.f. Fig. 1) for $K=1,\cdots,5$. For DNAs of the same number of hidden layers, we only use the DNA of the best performance achieved by cross-validation to generate a new representation; i.e., for the MFCC feature vector of a frame input to a subnet of this DNA, the output of neurons in $\mathcal{CS}$ at the code layer or layer K forms its new representation. SC evaluation described in Sect. 3.1.2 is carried out with such five representations corresponding to $K=1,\cdots,5$ respectively as well as MFCCs itself as a baseline, and all speakers and their utterances in four test corpora described in 3.1.1 are used respectively for SC evaluation.

It is observed from Fig. 2 that in general SC performance is improved monotonically as the number of hidden layers increases up to $K=4$ but appears to be degraded for $K=5$, comparing to $K=4$, for all four test corpora. In comparison to the MFCC baseline performance, it is evident from Fig. 2(b) and 2(d) that our DNA works better on corpora of noisy speech, NKING and RUS, given the fact that representations by our DNA always outperforms MFCCs regardless of the number of hidden layers. However, representations by our DNA does not performs better on WKING and CHN of clean speech until it has at least three hidden layers, as illustrated in Fig. 2(a) and 2(c), which is consistent with a well known fact that being used as a speaker-specific representation MFCCs appears adequate for clean speech but sensitive to noise without channel compensation [2], [19]. For the cross-corpora and/or the cross-language evaluation as depicted in Fig. 2, it is seen that the optimal performance is obtained by our DNA of a sufficient architecture depth of $K=4$ for all four corpora. In particular, an error-free performance is achieved by our DNA of seven hidden layers on RUS

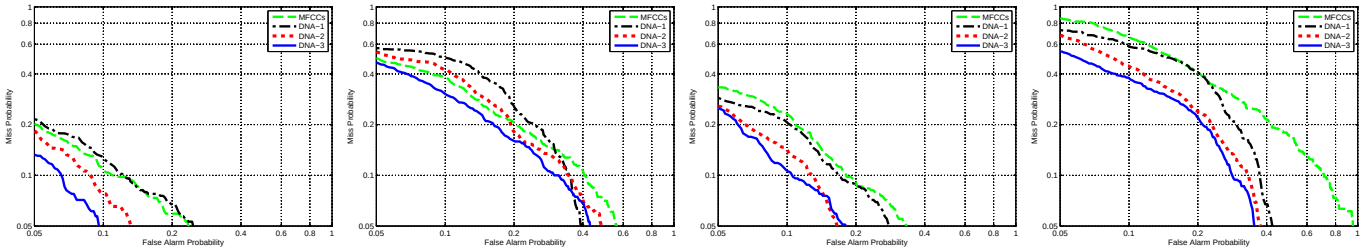

Fig. 3. SC performance (DET curves) of representations generated by our DNA trained on different data sets vs. MFCCs on WKING and NKING. (a) WKING (within-divide). (b) NKING (within-divide). (c) WKING (cross-divide). (d) NKING (cross-divide).

as evident in Fig. 2(d).

In summary, our experiments validate the importance of sufficient architecture depth in our DNA for good generalization in SSI extraction.

3.3 Validating the Role of Training Data

As a data-driven method, it is well known that training data critically determine the performance of a machine learning system. In this section, we present experiments to validate the role of training data in learning speaker-specific representations. Due to the limited space here, more results on investigating the importance of training data can be found from [44] and relevant issues will be discussed later on.

According to variabilities conveyed in data, we construct three training data sets based on all five training corpora described in 3.1.1 as follows: 1) TIMIT (speaker variability), 2) TIMIT, NTIMIT and SNTIMIT (speaker, land-line and cellular phone channel variabilities), and 3) all five training corpora (speaker, land-line and cellular phone channel, and handset variabilities). Based on our model selection results reported previously, we train DNAs of seven hidden layers (i.e., $K=4$) on three data sets, respectively. Hereinafter, we dub DNAs trained on the aforementioned three data sets *DNA-1*, *DNA-2* and *DNA-3*. Representations generated by three DNAs are used to build up $\mathcal{SM}$s against MFCCs in SC evaluation.

For test, we employ the LDC benchmark corpus KING as it covers nearly all possible variabilities faced by SR. The KING corpus consists of wide-band and narrow-band sets, WKING and NKING, and utterances of all speakers were recorded in 10 sessions in various environments [38]. Furthermore, there was a "great divide" between sessions 1-5 and 6-10; both recording device and environments changed, which alters spectral features of 26 speakers and leads to 10dB SNR reduction on average. Therefore, the KING provides an ideal test bed to evaluate the generalization capability. By using the same protocol introduced in [4], we conduct two experiments on WKING and NKING, respectively; i.e., *within-divide* where $\mathcal{SM}$s built on utterances in session 1 are compared to $\mathcal{SM}$s on those in sessions 2-5 and *cross-divide* where $\mathcal{SM}$s built on utterances in session 1 are compared with those in sessions 6-10.

In both with-divide and cross-divide experiments, it is observed from Fig. 3 that the use of training data conveying more variabilities always leads to better performance. In the within-divide setting, MFCCs outperforms representations by DNAs trained on data sets of fewer variabilities although the representation by DNA-3 always yields the best performance, as illustrated in Fig. 3(a) and 3(b). In contrast, representations by DNAs perform is generally superior to MFCCs in the cross-divide setting, as evident in Fig. 3(c) and 3(d). Once again, our results are consistent with the well known fact that MFCCs are sensitive to noise and environmental changes in encoding speaker-specific characteristics [2], [19]. Our experimental results validate the importance of training data; the training data must convey sufficient variabilities for good generalization in learning speaker-specific characteristics.

As illustrated in Fig. 3, DNA-3 trained on all five corpora as described in 3.1.1 yields the best representation that even outperforms those generated by a number of different deep architectures trained a subset of WKING or NKING in the same within-divide and cross-divide settings [34], [44], [45]. DNA-3 has a structure of 100, 100, 100 and 200 neurons in hidden layers 1-4 and $|\mathcal{CS}|$=100 in the code layer or hidden layer 4. In all the experiments reported in the rest of this paper, we shall use such a 100-dimensional feature vector by DNA-3 as a generic speaker-specific representation in comparison to several state-of-the-art SR techniques.

3.4 Vowel Distribution Visualization

Vowels have been recognized to be a main carrier of SSI [1], [2], [10], [11], [13], [15], [16]. TIMIT [38] provides phonetic transcription of all 10 utterances containing all 20 vowels in American English for every speaker. As all the vowels may appear in 10 different utterances, there are up to 200 vowel segments in length of 0.1-0.5 sec for every speaker, which enables us to investigate vowel distributions in a representation space for different speakers. As we model a speaker with the 1st- and the 2nd-order statistics on a speech segment, we treat both the mean vector and the covariance matrix (converted into one-dimensional vector) of a vowel speech segment as two feature vectors in terms of a specific representation. Hence, we visualize the mean and the covariance feature vectors of all different vowel segments for a speaker with the t-SNE method [46], a state-of-the-art non-linear visualization technique likely

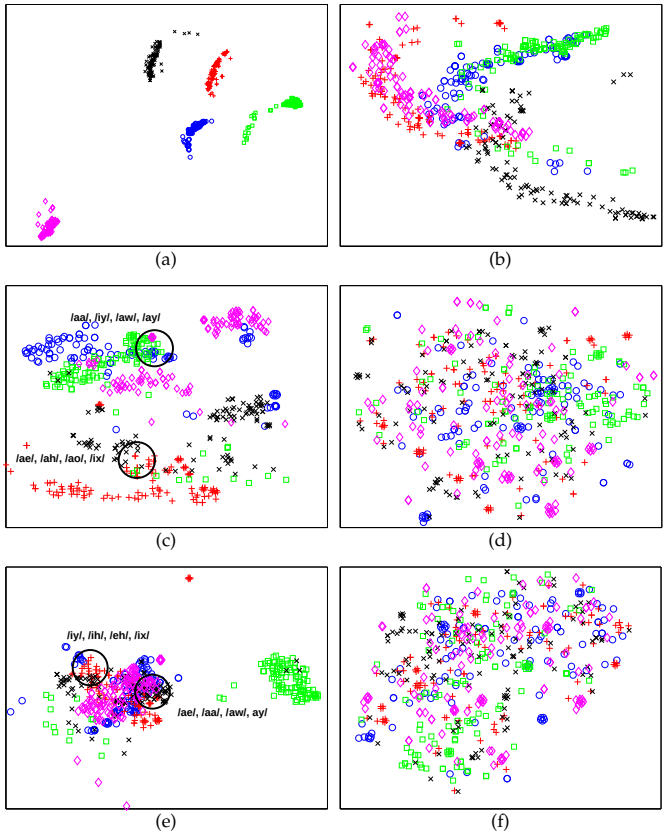

Fig. 4. Visualization of all 20 vowels in American English spoken by five speakers in terms of (a)-(b) $\mathcal{CS}$ representation, (c)-(d) $\overline{\mathcal{CS}}$ representation, and (e)-(f) MFCCs.

to reveal intrinsic manifolds, by projecting them onto a two-dimensional plane.

In the code layer of our DNA (c.f. Fig. 1), output of neurons 1-100 forms a speaker-specific representation, $\mathcal{CS}$, and that of remaining 100 neurons becomes a non-speaker related representation, $\overline{\mathcal{CS}}$. Thus, a vowel segment is characterized by a 100-dimensional mean and 10,000-dimensional covariance feature vectors for both $\mathcal{CS}$ and $\overline{\mathcal{CS}}$, respectively. For comparison, the vowel segment is also represented by 20-dimensional mean and 400-dimensional covariance MFCC feature vectors. For a noticeable effect, we randomly choose only five speakers (four females and one male) from the TIMIT validation set as described in Sect. 3.1.1 and visualize their vowel distributions in Fig. 4 in terms of $\mathcal{CS}$, $\overline{\mathcal{CS}}$ and MFCC representations, respectively, where a marker/color corresponds to a speaker. It is evident from Fig. 4(a) that, by using the $\mathcal{CS}$ mean vectors, most vowels spoken by a speaker are tightly grouped together while vowels spoken by different speakers are well separated. For the $\overline{\mathcal{CS}}$ mean vectors, closer inspection on Fig. 4(c) reveals that the same vowels spoken by different speakers are, to a great extent, co-located. Moreover, most of phonetically correlated vowels, as circled and labeled, are closely located in dense regions independent of speakers and genders. For comparison, we also visualize their MFCC mean vectors in Fig. 4(e) and observe that most of phonetically correlated vowels are also co-located, as circled and labeled, whilst others scatter across the plane and their positions are determined mainly by vowels but affected by speakers. In particular, most of vowels spoken by the male speaker, marked by □ and colored by green, are grouped tightly but isolated from those by all female speakers. In contrast, visualization of covariance feature vectors reveals less meaningful distributions due to a significant information loss incurred by dimension reduction. Nevertheless, it is still seen from Fig. 4(b) that those vowels spoken by the same speaker are distributed in some "manifolds" in terms of $\mathcal{CS}$ covariance feature vectors, although their separability is unclear. In Fig. 4(d), with $\overline{\mathcal{CS}}$ covariance feature vectors, vowels spoken by an individual seem to distribute evenly in the two-dimensional plane but there is no clear phoneme grouping. By MFCC covariance feature vectors, the distribution of vowels is depicted in Fig. 4(f) and we observe neither speaker nor phoneme groupings from this visualization.

In summary, the visualization in Fig. 4 demonstrates how our DNA learning extracts SSI and could also lend an evidence to justification on why MFCCs can be used in both SR and speech recognition [16]. It is also worth stating that visualization in Fig. 4, to a great extent, justifies why the original KL divergence used as a distance metric [1] does not work well for mono-Gaussian $\mathcal{SM}$s built on very short utterances but our modified KL distance metric in (11) often performs better in this situation [34], [44], [45].

4 Comparative Studies

In this section, we further evaluate our proposed approach by comparing it to several state-of-the-art techniques in terms of two typical yet real SR tasks, *speaker verification*, a supervised learning problem, and *speaker segmentation*, an unsupervised learning problem. In our experiments, once again, we use cross-corpora and/or cross-language protocols to assess generalization performance and mainly focus on the situation that a short utterance is available for SM as such a situation poses a greater challenge to SR [1], [2], [5],

4.1 Speaker Verification

Speaker verification (SV) is a process that accepts or rejects the identity claim of a speaker based on his/her voice and probably the commonest SR application scenario. Typically, a SV system works in supervised learning paradigm[3]; for an individual, his/her utterances as a reference are collected to build up his/her $\mathcal{SM}$ during learning and then the $\mathcal{SM}$ is used to test an utterance claimed to be spoken by the speaker. In our experiment, we compare ours described in Sect. 3.1.2 with two state-of-the-art techniques, *adapted GMM from a universal background model* (GMM-UBM) [6] and *convolutional deep belief network* (CDBN) [32], based on the Switchboard Cellular Part 2, a benchmark corpus that consists of 149 male and 207 female speakers and was used in the

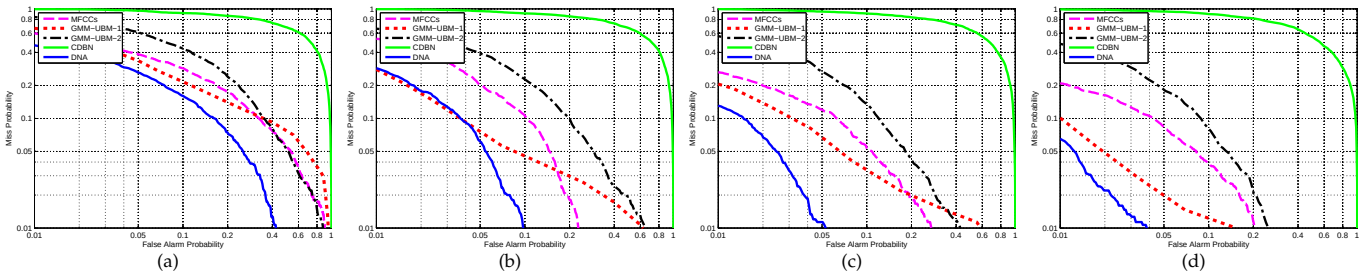

Fig. 5. SV performance (DET curves) of different methods on SRE03 with test utterances/segments of various lengths. (a) 1 sec. (b) 3 sec. (c) 5 sec. (d) 7 sec.

NIST Speaker Recognition Evaluation 2003 (SRE03) [38] as listed in TABLE 1. To better understand experimental results, MFCCs with the same SM and SC techniques used for our representation, as described in Sect. 3.1.2, are also employed as an essential baseline in this SV experiment.

The GMM-UBM trained on the MFCC representation has been generally recognized as one of the best SR techniques and yields state-of-the-art performance in different SR tasks [2], [6], [47]. The principle behind the GMM-UBM [6] is as follows: a UBM, a GMM of many Gaussian components, is created in advance based on a data set conveying different variabilities, e.g., a large speaker population and utterances collected from different channels, and then a GMM-based $\mathcal{SM}$ of an individual is built up with his/her short reference utterance(s) by adapting it from the UBM . It is well known that training data used for creating a UBM critically determines the performance of an adapted GMM [2], [6], [47], which is the same problem encountered in our DNA learning. In order to set baseline and evaluate generalization performance, we train two UBMs in terms of *close-set* and *open-set* settings. In the close-set setting, we assume that a speaker population is known and fixed and an SV system is developed only for this population. As a result, we use all 356 speakers' utterances with a duration of 90 sec per speaker in SRE03 to train a UBM. In the open-set setting, we assume that there is no *a prior* knowledge on speakers to be involved in SV and hence employ the exactly same training data used for our DNA-3 learning, as described in Sect. 3.1.1 and Sect. 3.3, to train a UBM. The standard EM algorithm is used to train two UBMs of 2048 Gaussian components on 20-order MFCCs. Then, individual speaker models are adapted using the MAP with the relevance factor equal to 16 from two UBMs, respectively, where only mean adaptation is used as suggested in [6]. Hereinafter, we name GMMs adapted from the close-set and the open-set UBMs *GMM-UBM-1* and *GMM-UBM-2*, respectively. Instead of evaluating the standard log-likelihood score with an adapted GMM $\mathcal{SM}$, the fast scoring procedure suggested in [6] is applied to produce a score for any test utterance/segment during recognition.

The CDBN was recently proposed to learn a generic speech representation and yields satisfactory performance in several audio classification tasks [32]. For a thorough evaluation, we also employ representations generated by the CDBN in our SV experiment. For a fair comparison, we strictly follow their experimental settings used in [32]; i.e., the same preprocessing procedure, the same CDBN structural parameters including the kernel size of feature maps and the neighborhood size for probabilistic maximal-pooling, and the sparsity penalty. While all other parameters are kept the same as used in [32], we also exhaustively search three tunable parameter values in a broad range for the best performance with the cross-validation method. In our experiment, we train the CDBN with the same training data used in our DNA-3 learning, as described in Sect. 3.1.1 and Sect. 3.3, and employ the exactly same SM and SC techniques used in our representation, as described in Sect. 3.1.2, for SV. As their CDBN structure has two hidden layers, output from either of hidden layers and their combination by concatenating output of two hidden layers form different representations [32]. In our SV experiment, we have investigated all three representations by the CDBN. In the sequel, we always use the best performance achieved by the CDBN to compare with others.

As there are only utterances of 2 min for each speaker in SRE03 and utterances of 90 sec per speaker have already been used to train the close-set UBM, we randomly divide the remaining utterances of 30 sec per speaker into two data sets: *reference* and *test* sets. The reference set contains utterances of 10 sec per speaker used to build up $\mathcal{SM}$s with different methods as described above for each individual speaker, and utterances of 20 sec per speaker in the test set are partitioned into short speech segments of different lengths ranging from 1 sec to 7 sec used as test speech. For every $\mathcal{SM}$, we follow the same test protocol used in [4], [5] to conduct 100 true speaker and 1775 imposter trials including all other speakers in this SV experiment. The same SV experiment was repeated with 3-fold cross-validation for reliability and average results are reported here. As our evaluation focuses on the effectiveness of a representation and the capacity of SM, again, we use the DET curve to show all possible errors in SV to avoid thresholding or DM issues encountered in a practical SV system [3].

It is evident from Fig. 5 that the our method generally outperforms all the others regardless of test lengths. For the GMM-UBM method, it is seen from Fig. 5. that its performance is critically determined by training data used for building up a UBM given the fact that the performance of GMM-UBM-1 is generally superior to

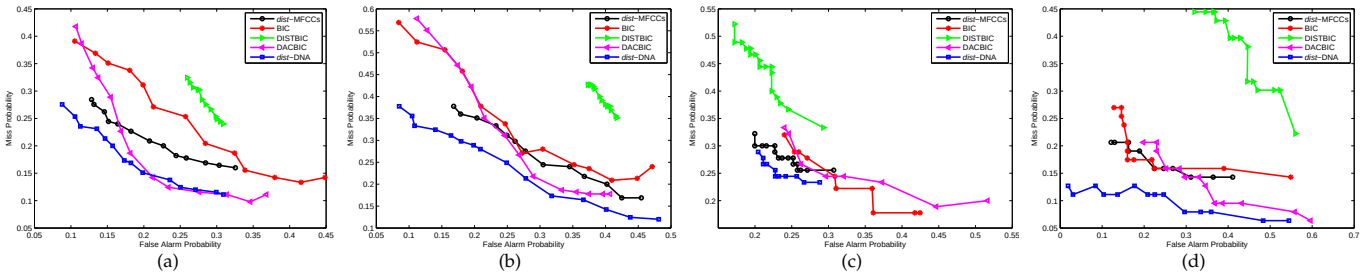

Fig. 6. SS performance (ROC curves) of different methods on four test data sets. (a) TIMIT. (b) NTIMIT. (c) CHN. (d) RT03.

that of others except ours and the baseline performance by mono-Gaussian $\mathcal{SM}$s on MFCCs for test length of 1 sec, as depicted in Fig. 5(a), but GMM-UBM-2 even underperforms the baseline performance for all test lengths, which confirms its well-known limitation of the GMM-UBM method [47]. From Fig. 5, it is observed that the CDBN always performs worse than all others including the essential baseline performance produced by mono-Gaussian $\mathcal{SM}$s on MFCCs. Our experimental results here indicate that the CDBN trained in an unsupervised learning style does not capture SSI well while it yields generic yet new speech representations that lead to satisfactory performance for various audio classification tasks [32]. In contrast, our DNA trained in a supervised learning style yields a generic speaker-specific representation that leads to better generalization in SSI extraction and favorable SV performance accordingly when short utterances are only available for SM and recognition. Relevant issues will be discussed later on.

4.2 Speaker Segmentation

Speaker segmentation (SS) is a task of detecting speaker change points in an audio stream and splitting it into acoustically homogeneous segments where every segment contains one speaker only [48]. As there is no *a prior* knowledge on an audio stream in general, e.g., speaker information and the number of speakers and change points, SS is a typical unsupervised learning task and an essential step for generic speaker diarization [49].

In our experiment, we compare ours described in Sect. 3.1.2, hereinafter dubbed *dist*-DNA, to both benchmark and state-of-the-art SS techniques including *distance-based* (*dist*-MFCCs) [48], *Bayesian information criterion* (BIC) [50], *distance-based BIC* (DISTBIC) [51] and *divide-and-conquer BIC* (DACBIC) [52] SS techniques where 20-order MFCC representation is used in all SS methods to be compared. In general, a distance-based method slides a window of the fixed size onto an audio stream to block it into short segments and two consecutive segments are always compared to determine if there is a possible speaker change point between the two segments. While BIC is applied to those BIC-based SS techniques for SC, the modified KL divergence in Eq. (11) as a distance metric is applied to all the distance-based methods for SC since very short segments are used for SM in SS.

To simulate conversations of short lengths in an audio stream, we adopted the same protocol used in [48], [51] to generate audio streams from three corpora of natural short utterances, TIMIT, NTIMIT and CHN, as listed in TABLE 1. With TIMIT and NTIMIT [38], we generate 50 audio streams totally and 25 for each. Each audio stream has a duration of about 40 sec and consists of 10 speaker segments of variable lengths ranging from 1.6 and 7.0 sec where a segment corresponds to a natural short utterance. Totally, utterances of 250 speakers, including all the speakers in the validation sets, from TIMIT and NTIMIT were used, respectively. Similarly, we created 10 audio streams with 50 speakers' utterances from the CHN corpus [40]. By concatenating utterances recorded in the same session, each audio stream of 45 sec on average consists of 10 speaker segments of variable lengths ranging from 3.0 to 5.0 sec . Furthermore, we also employed the data set of *ABC* broadcast news excerpts within a benchmark corpus used in NIST Rich Transcription Evaluation 2003 (RT03) to evaluate the SS performance. The total duration of news recording is 30 minutes in this data set where there are 70 speaker change points and the duration of a speaker segment is typically longer than 10 sec. The use of CHN and RT03 allows us to evaluate the generalization capability of our speaker-specific representation again with cross-language and/or cross-corpora settings.

In our experiment, we use a fixed-duration sliding window of 1.5 sec for SM in a distance-based method and allow a tolerance interval of 0.5 sec to validate a speaker change point. All other parameters are kept same for all methods. In an SS system, two types of errors are measured by the *false alarm rate* (FAR) and the *miss detection rate* (MDR) [48], which reflects recall and precision performance, respectively. As suggested in [52], we use a *receiver operating characteristic* (ROC) curve (FAR vs. MDR), which indicates typical errors made by different threshold settings to determine speaker change points, for performance evaluation.

It is evident from Fig. 6 that in general our *dist*-DNA outperforms all other methods on all four data sets. As a state-of-the-art SS method, the DACBIC [52] also achieves adequate results on all data set in general. As shown in Fig. 6(a), the DACBIC results in a comparable performance to ours overall but performs worse on the TIMIT data set in terms of the equal error rate when FAR equals MDR, a common performance index used in SR. Both the DACBIC and our *dist*-DNA generally outperform the baseline performance produced by *dist*-MFCCs on all data sets. In contrast, the original BIC [50], a benchmark SS technique, yields the comparable

performance to the baseline performance by *dist*-MFCCs on CHN and RT03 data sets, as shown in Fig. 6(c) and 6(d), but fails to do the same on TIMIT and NTIMIT data sets, as illustrated in Fig. 6(a) and 6(b). The DISTBIC [51], another benchmark SS technique, significantly underperforms other four methods including the baseline.

In summary, comparative studies suggest that our approach yields the robust performance on different types of data sets including short and long speaker segments and different variabilities. In particular, our approach using a simple speaker distance yields better performance than the state-of-the-art DACBIC of advanced divide-and-conquer mechanisms incorporated into BIC on MFCCs, which, once again, demonstrates strength and potential of a representation encoding SSI in SR.

5 Discussions

In this section, we discuss relevant issues and relate ours to previous work in terms of deep learning and SSI extraction.

Apart from experiments reported in this paper, we have done extensive experiments regarding our DNA learning in terms of SSI extraction [44]. Due to the limited space, we briefly summarize main outcomes below and details can be found from [44]. For validating the hybrid learning strategy [28], [29], we conducted experiments by a random initialization of DNA parameters without the pre-training described in Sect. 2.3.1. We found that the performance of the DNA is quite unstable without the pre-training and resultant representations lead to poorer SC performance than those yielded by the DNA trained with the hybrid strategy in our cross-validation experiments on a number of speech corpora. Besides results report in Sect. 3.2, other model selection experiments suggest that our DNA seems insensitive to structural parameters, e.g., DNAs of seven hidden layers but more hidden neurons in the code layer yields similar performance to that of the DNA-3 as described in Sect. 3.3. For validating training data of variabilities, we found that our DNA generates more robust representations when it is trained on a data set of more variabilities, e.g., using an additional corpus of emotional speech to train our DNA yields a more robust representation against emotional variabilities during test. It is also worth stating that the availability of computational resources and training data still limited our work reported in this paper. As model selection is computationally expensive, the number of candidate models had to be limited in our experiments. Due to a lack of data, the training set used in our work has yet to cover the ageing variability, a main mismatch in SR, although our method yields satisfactory results on test data of ageing variability as demonstrated in Sect. 3.3. Nevertheless, we notice that a heuristic algorithm for automatic model selection [53] was proposed very recently to facilitate deep learning. In our ongoing work, we shall be investigating such algorithms towards finding out the optimal DNA structure for a given training data set.

As described in Sect. 1, speech carries different yet mixed information but SSI is minor in comparison to predominant LI. Our empirical studies suggest that our success in SSI extraction is attributed to both unsupervised pre-training and supervised discriminative learning with a multi-objective loss. In particular, the use of data regularization in discriminative learning and distorted data in two learning phases plays a critical role in capturing intrinsic speaker-specific characteristics and variations caused by miscellaneous mismatches. Without discriminative learning, a DA trained with unsupervised learning only, e.g., the CDBN [32], tends to yield a new representation that redistributes different information in its encoding scheme but neither highlights minor SSI nor suppresses predominant LI given the fact that representations by the CDBN yield inadequate performance in SV, as demonstrated in Sect. 4.1, but works well for various audio classification tasks [32]. If we remove the data regularization term, $L_R(X_i;\Theta)$ defined in Eq.(3a), from the loss function in Eq. (3), our DNA is boiled down to a standard Siamese architecture [35]. Our results not reported here show that such an architecture yields a representation that often overfits to a training data set due to interference of predominant non-speaker related information [44], which does not seem to be a problem in predominant information extraction, e.g., facial identity information in a facial image. The previous work in face recognition [30] lends a clear evidence to support our argument where a Siamese DA without data regularization successfully captures predominant identity characteristics from facial images as, we believe, facial expression and other non-identity information conveyed in a facial image are minor in this situation. On the other hand, the use of distorted speech generated by adding channel noise to clean speech as additional training data further contributes to the success in SSI extraction. While the use of distorted data in the pre-training is in the same spirit of self-taught learning [54], we emphasize that the same distorted data are also used in discriminative learning to train our DNA. Our experiments not reported here reveal that the DNA trained on distorted data in both pre-training and discriminative learning considerately outperforms its counterpart where distorted data is merely used in its pre-training in terms of several different SR settings [44]. Hence, our work suggests that the idea of self-taught learning in an unsupervised learning setting [54] can be extended to supervised discriminative learning to facilitate robust feature learning in SSI extraction. In addition, it is worth stating that with the same settings, our DNA trained with the loss function defined in Eq. (3) are also distinctly superior to its counterpart trained with a loss function defined at the frame level in our previous work [34], which demonstrates the effectiveness of our loss function proposed in this paper.

In terms of architecture, our DNA resembles the one proposed in [55] for dimensionality reduction of handwritten digit images via learning a nonlinear embed-

ding. However, ours distinguishes itself from theirs in building blocks, loss functions and, more importantly, motivations. The DA proposed in [55] uses the restricted Boltzmann machines [28] as a building block to construct a deep belief subnet in their Siamese DA and the *neighborhood component analysis* (NCA) criterion [56] as their contrastive loss function to minimize the intra-class variability. However, the NCA does not meet our requirements as there are so many training examples in one class in our problem. Instead we propose a contrastive loss to minimize both intra- and inter-class variabilities simultaneously. On the other hand, intrinsic topological structures of a handwritten digit convey predominant information given the fact that without using the NCA loss a deep autoencoder has already yielded a satisfactory representation [27], [28], [31], [55]. Thus, the use of the NCA in [55] simply reinforces the topological invariance by minimizing other variabilities with a small amount of labeled data in the semi-supervised learning paradigm [55]. In our work, however, SSI is non-predominant in speech and hence a large amount of labeled data reflecting miscellaneous variabilities are demanded during supervised discriminative learning along with the unsupervised pre-training. Finally, our code layer yields an overcomplete representation to facilitate non-predominant information extraction. In contrast, a parsimonious representation seems more suitable for extracting predominant information since dimensionality reduction is likely to discover "principal" components that often associate with predominant information, as are evident in [30], [55].

In previous SR studies, some efforts [25], [26] were made for SSI extraction based on an assumption that the predominant LI or an speaker-independent speech representation is available or easy to extract. Then, the basic idea behind those methods is establishing a mapping from a general speech representation of mixed information to a speaker-specific representation for an individual speaker by highlighting SSI and suppressing LI simultaneously via discriminative learning. In order to achieve LI, they either simply assume that the low frequency sub-band in a spectral representation encodes only LI [26] or have to extract LI via learning to minimize the difference of each phoneme spoken by different speakers [25]. In contrast, our approach neither makes any assumption on the availability of LI nor explicitly use LI in SSI extraction. Although discriminative learning is used in both theirs and ours, our approach is towards extracting generic yet intrinsic SSI, a generic speaker-specific representation for any speakers, whilst the aforementioned methods [25], [26] merely establish a speaker-specific mapping for an individual. Therefore, our approach clearly distinguishes itself from the aforementioned methods in hypotheses and ultimate goals in terms of SSI extraction.

By means of functional magnetic resonance imaging techniques, recent neuroscience studies reveal that depending on a specific perceptual task, acoustic/speech information is processed in a hierarchical and selective way in human auditory cortex [57]. In particular, it has been discovered that processing of SSI is located in the right hemisphere while speech message processing takes place predominantly in the left hemisphere of brain [57]-[59]. This finding suggests that human selectively extracts SSI from speech by separating it from LI in an SR task and hence accomplishes various text-independent SR tasks effortlessly, which lends a strong evidence to support our methodology of using a hierarchical DNA with selective yet discriminative learning for SSI extraction. As demonstrated by the vowel distribution visualization, our speaker-specific representation can isolate SSI from LI, which are highly consistent with those achieved from neuroscience studies in terms of functionality. In a broader sense, our work presented in this paper suggests that *speech information component analysis* (SICA) becomes critical in various speech information processing tasks; the use of proper SICA techniques would result in task-specific speech representations to improve their performance radically. Our work on SSI extraction demonstrates that SICA is feasible via learning and, in particular, deep learning may be a promising methodology for SICA tasks.

6 Conclusions

In this paper, we have proposed an improved deep neural architecture and its learning algorithms for SSI extraction. Our empirical studies justify the importance of architecture depth and training data and demonstrate that SSI can be isolated from LI in general to facilitate text-independent SR. The further evaluation in both SV and SS tasks suggests that by incorporating a simple SM technique, the generic speaker-specific representation by our DNA leads to favorable performance in comparison to several state-of-the-art techniques. As a result, our approach paves an alternative way to improve SR performance. In our ongoing work, we shall be investigating alternative loss functions and structural learning algorithms to facilitate our DNA learning towards intrinsic SSI extraction as well as seeking suitable yet effective SM techniques to improve performance in different SR application scenarios.

Appendix

In this appendix, we derive the gradient of $L_D(X_1, X_2; \Theta)$, defined in Eq. (3b) in the main text, with respect to potentials, $\boldsymbol{u}_K(\boldsymbol{x}_{it})$, of neurons in the code layer to obtain Eq. (6) in the main text.

To simplify the presentation, we first elucidate our notation system that is completely consistent to that used in the main text. We collectively denote the output of neurons in $\mathcal{CS}$ at the code layer or layer K of subnet i (*i=1,2*) as $\mathcal{CS}(X_i) = \left\{\left(\left(\mathcal{CS}(\boldsymbol{x}_{it})\right)_j\right)_{j=1}^{|\mathcal{CS}|}\right\}_{t=1}^{T_B}$ for a speech segment of T_B frames, $X_i = \{\boldsymbol{x}_{it}\}_{t=1}^{T_B}$. Accordingly, we

have $\boldsymbol{\mu}^{(i)} = \big(\mu_j^{(i)}\big)_{j=1}^{|\mathcal{CS}|}$ and $\Sigma^{(i)} = \Big[\sigma_{ln}^{(i)}\Big]$ $(l,n{=}1,\cdots,|\mathcal{CS}|)$ where $\sigma_{ln}^{(i)} = \frac{1}{T_B-1}\sum_{t=1}^{T_B}\big[\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_l - \mu_l^{(i)}\big]\big[\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_n - \mu_n^{(i)}\big]^T$. In the following derivation, we also drop all explicit parameters in $L_D(X_1, X_2; \Theta)$ and rewrite it into $L_D = L_m + L_S$ where $L_m = \mathcal{I}D_m + (1-\mathcal{I})e^{-\frac{D_m}{\lambda_m}}$ and $L_S = \mathcal{I}D_S + (1-\mathcal{I})e^{-\frac{D_S}{\lambda_S}}$.

Using our notation described above, we immediately achieve

$$\begin{aligned}\frac{\partial L_D}{\partial \boldsymbol{u}_K(\boldsymbol{x}_{it})} &= \frac{\partial L_m}{\partial \boldsymbol{u}_K(\boldsymbol{x}_{it})} + \frac{\partial L_S}{\partial \boldsymbol{u}_K(\boldsymbol{x}_{it})} \\ &= \Big(\big([\mathcal{I} - \lambda_m^{-1}(1-\mathcal{I})e^{-\frac{D_m}{\lambda_m}}]\frac{\partial D_m}{\partial u_{Kj}(\boldsymbol{x}_{it})}\big)_{j=1}^{|\mathcal{CS}|}, \vec{\mathbf{0}}\Big) + \\ &\quad \Big(\big([\mathcal{I} - \lambda_S^{-1}(1-\mathcal{I})e^{-\frac{D_S}{\lambda_S}}]\frac{\partial D_S}{\partial u_{Kj}(\boldsymbol{x}_{it})}\big)_{j=1}^{|\mathcal{CS}|}, \vec{\mathbf{0}}\Big). \quad \text{(A.1)}\end{aligned}$$

Here, $\vec{\mathbf{0}}$ is a zero vector of $|\mathbf{h}_K| - |CS|$ elements corresponding to the gradients of the loss function $L_D(X_1, X_2; \Theta)$ with respect to potentials of all the neurons in $\overline{\mathcal{CS}}$ (c.f. Fig. 1), i.e., $\vec{\mathbf{0}} = (0)_{j=|CS|+1}^{|\mathbf{h}_K|}$. To facilitate the presentation, we define $\psi_j(\boldsymbol{x}_{it}) = \frac{\partial D_m}{\partial u_{Kj}(\boldsymbol{x}_{it})}$ and $\xi_j(\boldsymbol{x}_{it}) = \frac{\partial D_S}{\partial u_{Kj}(\boldsymbol{x}_{it})}$. Now we simply need to calculate $\psi_j(\boldsymbol{x}_{it})$ and $\xi_j(\boldsymbol{x}_{it})$ for $j{=}1,\cdots,|\mathcal{CS}|$ to obtain Eq. (6) in the main text.

As $D_m = ||\boldsymbol{\mu}^{(1)} - \boldsymbol{\mu}^{(2)}||_2^2 = \sum_{l=1}^{|\mathcal{CS}|}(\mu_l^{(1)} - \mu_l^{(2)})^2$, we have

$$\begin{aligned}\psi_j(\boldsymbol{x}_{it}) &= \frac{\partial D_m}{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}\frac{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}{\partial u_{Kj}(\boldsymbol{x}_{it})} \\ &= \frac{\partial\sum_{l=1}^{|\mathcal{CS}|}(\mu_l^{(1)} - \mu_l^{(2)})^2}{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}\frac{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}{\partial u_{Kj}(\boldsymbol{x}_{it})} \\ &= p_j^{(i)}\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big[1 - \big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big], \quad \text{(A.2)}\end{aligned}$$

where

$$p_j^{(i)} = \frac{\partial\sum_{l=1}^{|\mathcal{CS}|}(\mu_l^{(1)} - \mu_l^{(2)})^2}{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j} = \frac{2}{T_B}\text{sign}(1.5{-}i)(\mu_j^{(1)} - \mu_j^{(2)}),$$

and

$$\frac{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}{\partial u_{Kj}(\boldsymbol{x}_{it})} = \big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big[1 - \big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big]$$

given the fact that the transfer function used in the code layer or layer K is the sigmoid function. Collectively, we have $\boldsymbol{p}^{(i)} = \frac{2}{T_B}\text{sign}(1.5{-}i)(\boldsymbol{\mu}^{(1)} - \boldsymbol{\mu}^{(2)})$. Here, the notation $\text{sign}(1.5 - i)$ is introduced to simplify our presentation where i indicates subset i $(i = 1, 2)$.

Similarly, $D_S = ||\Sigma^{(1)} - \Sigma^{(2)}||_F^2 = \sum_{l=1}^{|\mathcal{CS}|}\sum_{n=1}^{|\mathcal{CS}|}(\sigma_{ln}^{(1)} - \sigma_{ln}^{(2)})^2$. Hence, we have

$$\begin{aligned}\xi_j(\boldsymbol{x}_{it}) &= \frac{\partial D_S}{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}\frac{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}{\partial u_{Kj}(\boldsymbol{x}_{it})} \\ &= \frac{\partial\sum_{l=1}^{|\mathcal{CS}|}\sum_{n=1}^{|\mathcal{CS}|}(\sigma_{ln}^{(1)} - \sigma_{ln}^{(2)})^2}{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}\frac{\partial\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j}{\partial u_{Kj}(\boldsymbol{x}_{it})} \\ &= q_j(\boldsymbol{x}_{it})\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big[1 - \big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_j\big], \quad \text{(A.3)}\end{aligned}$$

where

$$q_j(\boldsymbol{x}_{it}){=}\frac{4}{T_B{-}1}\text{sign}(1.5{-}i)\sum_{n=1}^{|\mathcal{CS}|}(\sigma_{jn}^{(1)}{-}\sigma_{jn}^{(2)})\big[\big(\mathcal{CS}(\boldsymbol{x}_{it})\big)_n{-}\mu_n^{(i)}\big],$$

and, collectively, we have $\boldsymbol{q}(\boldsymbol{x}_{it}){=}\frac{4}{T_B-1}\text{sign}(1.5{-}i)(\Sigma^{(1)}{-}\Sigma^{(2)})[\mathcal{CS}(\boldsymbol{x}_{it}) - \boldsymbol{\mu}^{(i)}]$.

Inserting Eqs. (A.2) and (A.3) into Eq. (A.1), we obtain Eq. (6) in the main text as

$$\begin{aligned}\frac{\partial L_D}{\partial \boldsymbol{u}_K(\boldsymbol{x}_{it})} &= \Big(\big([\mathcal{I} - \lambda_m^{-1}(1-\mathcal{I})e^{-\frac{D_m}{\lambda_m}}]\psi_j(\boldsymbol{x}_{it})\big)_{j=1}^{|\mathcal{CS}|}, \vec{\mathbf{0}}\Big) + \\ &\quad \Big(\big([\mathcal{I} - \lambda_S^{-1}(1-\mathcal{I})e^{-\frac{D_S}{\lambda_S}}]\xi_j(\boldsymbol{x}_{it})\big)_{j=1}^{|\mathcal{CS}|}, \vec{\mathbf{0}}\Big).\end{aligned}$$

ACKNOWLEDGMENTS

Authors would like to thank G. Hinton, H. Larochelle and H. Lee for personal communication. Authors are also grateful to H. Lee for providing their CDBN [32] code and L. Wang for providing their Chinese corpus [40], both of which were used in our experiments. An extended abstract of this manuscript was presented in NIPS'11 [45]. K. Chen is the corresponding author.

REFERENCES

[1] J. Campbell, "Speaker recognition: A tutorial," *Proceedings of the IEEE*, vol. 85, pp. 1437-1462, 1997.
[2] D. Reynolds and W. Campbell, "Text-independent speaker recognition," in *Handbook of Speech Processing*, J. Benesty et al (Eds.), Berlin: Springer, pp. 763-781, 2008.
[3] K. Chen, "Towards better making a decision in speaker verification," *Pattern Recognition*, vol. 36, pp. 329-346, 2003.
[4] D. Reynolds, "Speaker Identification and verification using Gaussian mixture speaker models," *Speech Communication*, vol. 17, pp. 91-108, 1995.
[5] D. Reynolds and R. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. Speech Audio Process.*, vol. 3, pp. 72-83, 1995.
[6] D. Reynolds, T. Quatieria, and R. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Process.*, vol. 10, pp. 19-41, 2000.
[7] W. Campbell, J. Campbell, D. Reynolds, E. Singer, and P. Torres-Carrasquillo, "Support vector machines for speaker and language recognition," *Comput. Speech Lang.*, vol. 20, pp. 210–229, 2006.
[8] B. Fauve, D. Matrouf, N. Sheffer, J. Bonastre, and J. Mason, "State-of-the-art performance in text-independent speaker verification through open-source software," *IEEE Trans. Audio Speech Lang. Process.*, vol. 15, pp. 1960-1968, 2007.
[9] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Trans. Audio Speech Lang. Process.*, vol. 15, pp. 1435-1447, 2007.
[10] M. Joos, "Acoustic phonetics," *Language*, vol. 24, pp. 1-136, 1948.
[11] P. Ladefoged and D. Broadbent, "Information conveyed by vowels," *J. Acoust. Soc. Am.*, vol. 29, pp. 98-104, 1957.
[12] L. Welling, H. Ney, and S. Kanthak, "Speaker adaptive modeling by vocal track normalization," *IEEE Trans. Audio Speech Process.*, vol. 10, pp. 415-426, 2002.
[13] K. Johnson, "Speaker normalization in speech perception," in *Handbook of Speech Perception*, D. Benesty and R. Remez (Eds.), Oxford: Blackwell Publishing, pp. 363-389, 2005.
[14] R. Turner, T. Walters,J. Monaghan, and R. Patterson, "A statistical formant-pattern model for estimating vocal-tract length from formant frequency data," *J. Acoust. Soc. Am.*, vol. 125, pp. 2374-2386, 2009.
[15] B. Moore, *An Introduction to Psychology of Hearing (5th Edition)*, San Diego: Elsevier Academic Press, 2003.

[16] X. Huang, A. Acero, and H. Hon, *Spoken Language Processing*, New York: Prentice Hall, 2001.

[17] L. Deng, Y. Dong, and A. Acero, "Structured speech modeling," *IEEE Trans. Audio Speech Lang. Process.*, vol. 14, pp. 1492-1504, 2006.

[18] D. O'Shaughnessy, "Automatic speech recognition: history, methods and challenges," *Pattern Recognition*, vol. 36, pp. 2965-2975, 2008.

[19] R. Mammone, X. Zhang, and R. Ramachandran, "Robust speaker recognition: a feature-based approach," *IEEE Signal Process. Magazine*, vol. 13, pp. 58-71, 1996.

[20] B. Atal, "Automatic speaker recognition based on pitch contours," *J. Acous. Soc. Amer.*, vol. 52, pp. 1688-1697, 1972.

[21] Y. Long, Z. Yan, F. Soong, L. Dai, and W. Guo, "Speaker characterization using spectral subband energy ratio based on harmonic plus noise model," in *Proc. ICASSP*, 2011.

[22] G. Doddington, "Speaker recognition based on idiolectal differences between speakers," In *Proc. Eurospeech*, 2001.

[23] B. Peskin, J. Navratil, J. Abramson, D. Jones, D. Klusacek, D. Reynolds, and B. Xiang, "Using prosodic and conversational features for high-performance," in *Proc. ICASSP*, vol. 4, 2003.

[24] G. Jang, T. Lee, and Y. Oh, "Learning statistically efficient feature for speaker recognition," in *Proc. ICASSP*, 2001.

[25] N. Malayath, N. Hermansky, S. Kajarekar, and B. Yegnanarayana, "Data-driven temporal filters and alternatives to GMM in speaker verification," *Digital Signal Process.*, vol. 10, pp. 55-74, 2000.

[26] H. Misra, I. Shajith, and B. Yegnanarayana, "Speaker-specific mapping for text-independent speaker recognition," *Speech Communication*, vol. 39, pp. 301-310, 2003.

[27] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504-507, 2006.

[28] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp. 1527-1554, 2006.

[29] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, pp. 1-127, 2009.

[30] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," In *Proc. IEEE CVPR*, 2005.

[31] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks," *J. Machine Learning Res.*, vol. 17, pp. 1-40, 2009.

[32] H. Lee, Y. Largman, P. Pham, and A. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in Neural Information Processing Systems*, vol. 22, 2009.

[33] A. Mohamed, G. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. Audio Speech Lang. Process.*, vol. 20, pp. 14-22, 2012.

[34] K. Chen and A. Salman, "Learning speaker-specific characteristics with a deep neural architecture," *IEEE Trans. Neural Networks*, vol. 22, pp. 1744-1756, 2011.

[35] J. Bromley, I. Guyon, Y. LeCun, E. Sackinger, and R. Shah, "Signature verification using a Siamese time delay neural network," in *Advances in Neural Information Processing Systems*, vol. 5, 1993.

[36] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "Energy-based models," in *Predicting Structured Outputs*, G. Bakir et al (Eds.), Cambridge, MA: MIT Press, pp. 191-246, 2007.

[37] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders,"in *Proc. AISTATS*, 2007.

[38] Linguistic Data Consortium (LDC). [online] www.ldc.upenn.edu

[39] J. Proakis, "*Digital Communications (4th Ed.)*," New York: McGraw-Hill, 2001.

[40] L. Wang, "Chinese Speech Corpus," Tech. Rep., *SIAT-CAS*, 2008.

[41] Russian Speech Corpus. [online] www.repository.voxforge1.org

[42] F. Bimbot, I. Magrin-Chagnolleau, and L. Mathan, "Second-order statistical methods for text-independent speaker identification.," *Speech Communications*, vol. 17, pp. 177-192, 1996.

[43] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Pryzbocki, "The DET curve in assessment of detection task performance," in *Proc. Eurospeech*, 1997.

[44] A. Salman, "Learning speaker-specific characteristics with deep neural architectures," PhD Thesis, School of Computer Science, The University of Manchester, 2012.

[45] K. Chen and A. Salman, "Extracting speaker-specific information with a regularized Siamese deep network," in *Advances in Neural Information Processing Systems*, vol. 24, 2011.

[46] L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE", *J. Machine Learning Res.*, vol. 9, pp. 2579-2605, 2008.

[47] T. Hason and J. Hansen, "A study on universal background model training in speaker verification," *IEEE Trans. Audio Speech Lang. Process.*, vol. 19, pp. 1890-1898, 2011.

[48] M. Kotti, V. Moschou, and C. Kotropoulos, "Speaker segmentation and clustering," *Signal Processing*, vol. 88, pp. 1091-1124, 2008.

[49] S. Tranter and D. Reynolds, "An overview of automatic speaker diarization systems," *IEEE Trans. Audio Speech Lang. Process.*, vol. 14, pp. 1557-1564, 2006.

[50] S. Chen and P. Gopalakrishnan, "Speaker environment and channel change detection and clustering via Bayesian information criterion," in *Proc. DAPRA Speech Recognition Workshop*, 1998.

[51] P. Delacourt and C. Wellekens, "DISTBIC: a speaker-based segmentation for audio data indexing," *Speech Communication*, vol. 32, pp. 111-126, 2000.

[52] S. Cheng, H. Wang, and H. Fu, "BIC-based speaker segmentation using divide-and-conquer strategies with application to speaker diarization," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, pp. 141-157, 2010.

[53] G. Zhou, K. Sohn, and H. Lee, "Online incremental learning with denoising autoencoders," in *Proc. AISTATS*, 2012.

[54] R. Raina, A.Battle, H. Lee, B. Packer, and A. Ng, "Self-taught learning: transfer learning from unlabeled data," in *Proc. ICML*, 2007.

[55] R. Salakhutdinov and G. Hinton, "Learning a non-linear embedding by preserving class neighbourhood structure," in *Proc. AISTATS*, 2007.

[56] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov, "Neighbourhood component analysis," in *Advances in Neural Information Processing Systems*, vol. 17, 2005.

[57] P. Belin, R. Zatorre, P. Lafaille, P. Ahad, and B. Pike, "Voice-selective areas in human auditory cortex," *Nature*, vol. 403, pp. 309-312, 2000.

[58] P. Wong, H. Nusbaum, and P. Small, "Neural bases of talker normalization," *J. Cogn. Neurosci.*, vol. 16, pp. 1173-1184, 2004.

[59] K. von Kriegstein, D. Smith, R. Patterson, S. Kiebel, and T. Griffiths, "How the human brain recognizes speech in the context of changing speakers," *J. Neurosci.*, vol. 30, pp. 629-638, 2010.