Speech Communication 118 (2020) 10–20

Contents lists available at ScienceDirect

Speech Communication

journal homepage: www.elsevier.com/locate/specom

Cosine metric learning based speaker verification☆

Zhongxin Bai[a,b], Xiao-Lei Zhang[a,b,*], Jingdong Chen[a,b]

[a] Research & Development Institute of Northwestern Polytechnical University in Shenzhen, Shenzhen, China
[b] Center of Intelligent Acoustics and Immersive Communications (CIAIC) and the School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China

ARTICLE INFO

Keywords:
Cosine metric learning
Inter-session variability compensation
Speaker verification

ABSTRACT

The performance of speaker verification depends on the overlap region of the decision scores of true and imposter trials. Motivated by the fact that the overlap region can be reduced by maximizing the between-class distance while minimizing the within-class variance of the trials, we present in this paper two cosine metric learning (CML) back-end algorithms. The first one, named m-CML, aims to enlarge the between-class distance with a regularization term to control the within-class variance. The second one, named v-CML, attempts to reduce the within-class variance with a regularization term to prevent the between-class distance from getting smaller. The regularization terms in the CML methods can be initialized by a traditional channel compensation method, e.g., the linear discriminant analysis. These two algorithms are combined with front-end processing for speaker verification. To validate their effectiveness, m-CML is combined with an i-vector front-end since it is good at enlarging the between-class distance of Gaussian score distributions while v-CML is combined with a d-vector or x-vector front-end as it is able to reduce the within-class variance of heavy-tailed score distributions significantly. Experimental results on the NIST and SITW speaker recognition evaluation corpora show that the proposed algorithms outperform their initialization channel compensation methods, and are competitive to the probabilistic linear discriminant analysis back-end in terms of performance. For comparison, we also applied the m-CML and v-CML methods to the i-vector and x-vector front-ends.

1. Introduction

Speaker verification is a task of verifying whether the claimed identity of a test speaker is true or not. A speaker verification system, in general, can be divided into two parts: a front-end, which extracts identity features given a speech signal (an utterance), and a back-end, which verifies the claim of a test speaker in a trial by calculating the similarity score of the identity features of the test and enrollment speakers of the trial (Dehak et al., 2011; Li and Mak, 2015; Hasan and Hansen, 2014).

In the literature, three front-ends have been widely studied: i-vector front-end based on the Gaussian mixture model and the universal background model (GMM-UBM) (Reynolds et al., 2000), i-vector front-end based on deep neural network and UBM (DNN-UBM), and the deep embedding based front-end, such as d-vector (Variani et al., 2014) and x-vector (Snyder et al., 2017; 2018). The GMM-UBM based i-vector front-end first models all speakers by a single GMM, and then uses factor analysis (Dehak et al., 2011) to reduce the dimension of the supervectors (Campbell et al., 2006b; 2006a; Kinnunen and Li, 2010) produced by GMM-UBM. The DNN-UBM/i-vector front-end (Lei et al., 2014; Kenny et al., 2014; Richardson et al., 2015a; Campbell, 2014) uses the posterior probability of senones produced by the acoustic model of a speech recognition system to replace the posterior probability produced by GMM-UBM at the frame level. The deep embedding based front-end takes the activations of the last hidden layer or bottleneck layer as the speaker features (Variani et al., 2014; Richardson et al., 2015b; Wang et al., 2017; Yaman et al., 2012; Snyder et al., 2017; 2018).

In comparison, commonly used back-ends include the cosine similarity scoring (Dehak et al., 2011), the support vector machine (Cumani and Laface, 2014), and the probabilistic linear discriminant analysis (PLDA) (Prince and Elder, 2007; Kenny, 2010; Garcia-Romero and Espy-Wilson, 2011; Li et al., 2017; Mak et al., 2016). Those back-ends are linear or shallow classifiers, which contain either no or at most one nonlinear layer. To better capture the nonlinearity of data, DNNs

☆ This work was supported in part by the National Key Research and Development Program of China under Grant No. 2018AAA0102200 and in part by the Key Program of National Science Foundation of China (NSFC) Grant No. 61831019 and the NSFC and Israel Science Foundation (ISF) joint research program under Grant No. 61761146001, in part by the NSFC funding scheme under grant No. 61671381, in part by the Project of the Science, Technology, and Innovation Commission of Shenzhen Municipality under grant No. JCYJ20170815161820095, and in part by the Shaanxi Natural Science Basic Research Program under grant No. 2018JM6035.

* Corresponding author at: Center of Intelligent Acoustics and Immersive Communications (CIAIC) and the School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China.

E-mail addresses: zxbai@mail.nwpu.edu.cn (Z. Bai), xiaolei.zhang@nwpu.edu.cn (X.-L. Zhang), jingdongchen@ieee.org (J. Chen).

https://doi.org/10.1016/j.specom.2020.02.003
Received 18 April 2019; Received in revised form 23 December 2019; Accepted 17 February 2020
Available online 20 February 2020

with more than one nonlinear layers have been investigated for back-ends in various ways, e.g. (Stafylakis et al., 2012; Senoussaoui et al., 2012; Ghahabi and Hernando, 2014; 2017; Tan et al., 2018). For example, different combinations of restricted Boltzmann machines were studied in Stafylakis et al. (2012) and Senoussaoui et al. (2012). Ghahabi *et al.* took deep belief networks (DBNs) as a discriminative back-end (Ghahabi and Hernando, 2014), and further proposed a hybrid system based on DBN and DNN to discriminatively model each target speaker (Ghahabi and Hernando, 2017). Tan et al. (2018) applied DNN to compensate decision score shifts caused by strong additive noise.

Because the output of a front-end is both inter-session-dependent and speaker-dependent, it is necessary to compensate channel- and session-variability before scoring. Common compensation methods include linear discriminant analysis (LDA) (Bishop, 2006), within-class covariance normalization (WCCN) (Hatch et al., 2006), and nuisance attribute projection (NAP) (Campbell et al., 2006b), which are all linear transformations. Recently, nonlinear compensation methods have drawn much attention. For example, Cumani et al. (2017) proposed a nonlinear transformation of i-vectors to make them more suitable for the PLDA back-end. Zheng Tieran (2018) developed a DNN based dimensionality reduction model as an alternative to LDA. Reported results showed that those back-ends can generate reasonably good performance. However, those back-end were developed without directly considering the evaluation metrics of speaker verification, such as equal error rate (EER), their performance may be still suboptimal.

Two classes of methods have been investigated to optimize the evaluation metrics directly. The first one, named *back-end metric learning*, uses objective functions that are equivalent to the evaluation metrics. For example, the work in Ahmad et al. (2014) and Li et al. (2016) attempts to maximize the margin between target and imposter trials with the triplet loss. Fang et al. (2013), also employed neighborhood component analysis to learn a projection matrix, which minimizes the average leave-one-out k-nearest neighbor classification error. The other class, based on the so-called *end-to-end deep learning*, jointly trains the front-end and back-end of DNN based systems. More specifically, these methods for text-dependent speaker verification (Heigold et al., 2016; Wan et al., 2017) learn a deep model, which maps a pair of enrollment and test utterances directly to a cosine similarity score. In Snyder et al. (2016), David et al. applied a similar end-to-end framework to jointly train a deep neural network front-end and a PLDA-like back-end. Both classes of methods have demonstrated promising results, indicating that directly optimizing the evaluation metrics like criteria is a proper approach to improving performance of speaker verification. However, further research is indispensable to unveil the full potential of this approach.

This paper is devoted to the problem of speaker versification with its focal point placed on the back-end metric learning under the following two considerations. First, a metric learning based back-end can be combined flexibly with a front-end. Second, a back-end metric learning method may be easily extended to an end-to-end deep learning method, as if it is optimized by a gradient descent algorithm. We propose a *cosine metric learning* (CML) framework based on cosine similarity scoring. CML aims to improve EER directly by minimizing the overlap region of the decision scores between true and imposter trials. We present two algorithms, leading to two back-ends, both learn a linear transformation through gradient descent. The first back-end, named *m-CML*, aims to enlarge the between-class distance with a regularization term to control the within-class variance. We apply this back-end to an i-vector front-end since it is good at enlarging the between-class distance of Gaussian score distributions. The second back-end, called *v-CML*, attempts to reduce the within-class variance with a regularization term to control the between-class distance. This back-end is combined with a d-vector front-end since it is able to reduce the within-class variance of heavy-tailed score distributions significantly. The regularization terms in the proposed methods play a very important role on performance, which can be initialized by a linear transform produced from a traditional channel compensation method, such as linear discriminant analysis (LDA) (Bishop, 2006), within-class covariance normalization (WCCN) (Hatch et al., 2006), nuisance attribute projection (NAP) (Campbell et al., 2006b), etc. The presented methods are evaluated on the NIST speaker recognition evaluation (SRE) corpora. Results demonstrate that these methods can combine traditional compensation methods and the cosine similarity scoring effectively for optimizing EER.

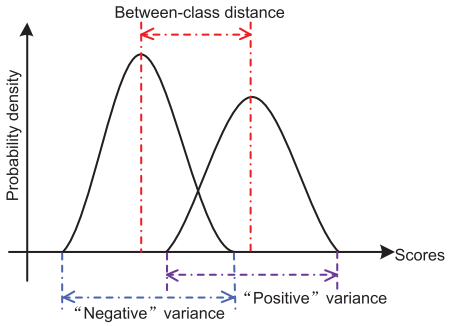

Fig. 1. Probability distribution of the decision scores produced from a speaker verification system. "Positive" denotes the true trials, "Negative" denotes the imposter trials.

This paper differs from our preliminary work (Bai et al., 2018) in several major aspects, which include a new v-CML algorithm in this paper, a faster and more stable optimization algorithm of L-BFGS, a further analysis of the variation of the between-class distance and within-class variance, and the computational complexity analysis of the CML. Consequently, experimental results in this paper are different from those reported in Bai et al. (2018).

The rest of this paper is organized as follows. We discuss the motivation and problem formulation in Section 2. Section 3 presents the theory and optimization algorithms for cosine metric learning. Experiments are conducted and the results are presented in Section 4. Finally, Section 5 gives some important conclusions.

2. Motivation

The objective of this work is to optimize EER of a speaker verification system given a well-trained front-end. It is known that speaker verification is a two class classification problem—true trials and imposter trials, and EER is determined by the overlap region of the decision scores of the two classes, as illustrated in Fig. 1. It is seen from the figure that the overlap region of the decision scores relies heavily on the between-class distance and within-class variance. Consequently, optimizing EER can be transformed into an equivalent problem of enlarging the between-class distance while reducing the within-class variance.

To study back-end metric learning, we adopt three representative front-ends: the GMM-UBM/i-vector front-end, the d-vector front-end, and the x-vector front-end. The first one contains a GMM-UBM (Reynolds et al., 2000), which is trained from the pool of all speech frames of the development data, and a total variability matrix (Dehak et al., 2011) that encompasses both speaker- and channel-variability. It first extracts a supervector from each utterance, which is the zero- and first-order Baum-Welch statistics of the utterance, and then reduces the supervectors to low-dimensional i-vectors by the total variability matrix. In comparison, the d-vector front-end (Variani et al., 2014) averages the frame-level features of an utterance produced from the top hidden layer of a DNN classifier for an utterance-level d-vector. The DNN is trained to minimize the classification error of speech frames, where the ground-truth label of a speech frame is the indicator vector of the speaker to whom the speech frame belongs to. The x-vector (Snyder et al., 2017; 2018) front-end adopts a similar framework with the d-vector front-end except that it introduced a statistic pooling layer into DNN to project variable-length segments into embedding vectors. However, the feature vectors produced from the three front-ends exhibit very

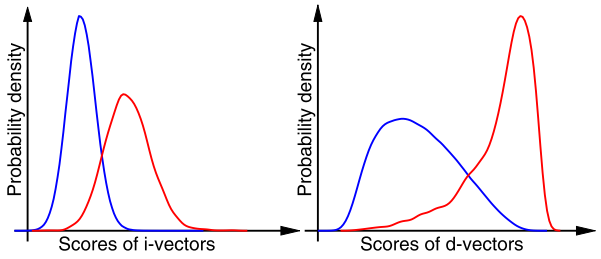

Fig. 2. Probability distribution of the cosine similarity scores of i-vectors and d-vectors.

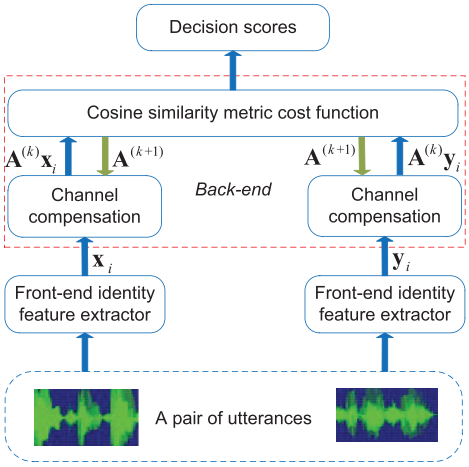

Fig. 3. Diagram of the cosine metric learning back-end based system.

different statistical properties. As illustrated in Fig. 2, if the cosine similarity scoring is adopted as the back-end, the probability distribution of the decision scores produced from a GMM-UBM/i-vector front-end approximates a normal distribution, and the overlap region of the scores depends mainly on the between-class distance. But the probability distribution of the decision scores produced from a d-vector or x-vector front-end is heavy-tailed (Variani et al., 2014), where the overlap region may be reduced significantly by reducing the within-class variance.

Motivated by the above facts, as well as the experimental results from Dehak et al. (2011) that cosine similarity scoring is good enough in many cases, we present in this paper two back-end metric learning methods based on cosine similarity scoring. The first one aims to enlarge the between-class distance while controlling the within-class variance to be small. The second one attempts to reduce the within-class variance while maintaining the between-class distance unchanged.

3. Cosine metric learning based back-ends

Let us first provide an overview of the proposed CML based speaker verification system and then present the objective functions and optimization algorithm of two CML back-ends.

3.1. System overview

The proposed cosine metric learning based speaker verification system is illustrated in Fig. 3. In the development stage, we first train a front-end, and extract an identity feature vector $\mathbf{x}$ from each training utterance by the specified front-end. In this work, we adopt the i-vector and d-vector based front-ends though many front-ends can be used. Then, the true and imposter trials are constructed manually from the identity feature vectors, due to the requirement for training the proposed CML back-ends, where a down-sampling method for constructing the training set will be given in Section 4.2.2). Finally, we train a CML back-end, which produces a linear transformation matrix $\mathbf{A}$. The objective function of the CML back-end contains a regularization term. This term should be initialized from an initial compensation transformation matrix $\mathbf{A}_0$, which can be produced from an existing channel or session compensation technique, e.g., LDA, WCCN, etc.

In the test stage, we extract the identity feature vectors of a test trial from the front-end, and then conduct verification by the cosine similarity scoring as follows:

$$S(\mathbf{x}_{\text{enroll}}, \mathbf{x}_{\text{test}}; \mathbf{A}) = \frac{\langle \mathbf{A}\mathbf{x}_{\text{enroll}}, \mathbf{A}\mathbf{x}_{\text{test}} \rangle}{\|\mathbf{A}\mathbf{x}_{\text{enroll}}\| \|\mathbf{A}\mathbf{x}_{\text{test}}\|} \gtrless \theta. \tag{1}$$

where $\mathbf{x}_{\text{enroll}}$ and $\mathbf{x}_{\text{test}}$ denote, respectively, the identity feature vectors of the enrollment and test utterances of the trial, and θ is a decision threshold.

3.2. Objective functions

In the development stage, we first construct a development set $\{(\mathbf{x}_i, \mathbf{y}_i, l_i)\}_{i=1}^N$ from the identity feature vectors of training utterances, as illustrated in Fig. 3, where $(\mathbf{x}_i, \mathbf{y}_i, l_i)$ is a training trial and N is the number of the training trials with $\mathbf{x}_i$ and $\mathbf{y}_i$ as a pair of the identity feature vectors and l_i as the ground-truth label of the trial. If $\mathbf{x}_i$ and $\mathbf{y}_i$ are from the same speaker, we set $l_i = 1$; otherwise we set $l_i = -1$. We define the index sets of the true and imposter trials as $\text{pos} = \{i | l_i = 1\}_{i=1}^N$ and $\text{neg} = \{i | l_i = -1\}_{i=1}^N$, respectively.

The optimization problem of CML is given by:

$$\mathbf{A}^* = \arg\min_{\mathbf{A}} f_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) \tag{2}$$

with $f_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i)$ defined as

$$f_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) = \ell(\{(\mathbf{x}_i, \mathbf{y}_i, l_i)\}_i; \mathbf{A}) + \lambda \Omega_{f_{\mathbf{A}}} \tag{3}$$

where $\ell(\cdot)$ is a loss function that minimizes EER under the cosine similarity scoring $S(\cdot)$, $\Omega_{f_{\mathbf{A}}}$ is a regularization term controlling the complexity of the learning machine, $\lambda > 0$ is a tunable hyperparameter, and $f_{\mathbf{A}}(\cdot) = \ell(\cdot) + \lambda \Omega_{f_{\mathbf{A}}}$ is the learning machine. Because parameter $\mathbf{A}$ is a linear transformation matrix, we adopt a common regularization term $\Omega_{f_{\mathbf{A}}} = \|\mathbf{A}\|_2^2$. As will be discussed later, the loss function $\ell(\cdot)$ under the cosine similarity scoring framework is nonconvex; therefore, we add an initialization $\mathbf{A}_0$ to $\Omega_{f_{\mathbf{A}}}$ so as to prevent from bad local minima:

$$\Omega_{f_{\mathbf{A}}} = \|\mathbf{A} - \mathbf{A}_0\|_2^2 \tag{4}$$

where $\mathbf{A}_0$ is a good linear model produced from any channel or session compensation method, see Section 3.2.3 for the compensation techniques in consideration. When $\lambda \to \infty$, $\mathbf{A}$ approaches to $\mathbf{A}_0$. In other words, the performance of the proposed method is lower-bounded by its initialization technique.

In the following, we introduce two loss functions $\ell(\cdot)$, which aims to enlarge the between-class distance and reduce the within-class variance, respectively.

3.2.1. m-CML: an objective for enlarging between-class distance

It is easy to see that the means of the decision scores of the true and imposter trials are $\frac{1}{|\text{pos}|}\sum_{i\in\text{pos}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A})$ and $\frac{1}{|\text{neg}|}\sum_{i\in\text{neg}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A})$ respectively. Hence, a natural choice for enlarging the between-class distance is to solve the following optimization problem:

$$\arg\max_{\mathbf{A}} \frac{1}{|\text{pos}|}\sum_{i\in\text{pos}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) - \frac{1}{|\text{neg}|}\sum_{i\in\text{neg}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) \tag{5}$$

which can be changed to minimize the following function:

$$\ell = -\sum_{i\in\text{pos}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) + \alpha \sum_{i\in\text{neg}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) \tag{6}$$

where $\alpha = \frac{|\text{pos}|}{|\text{neg}|}$. Note that α may also be defined as a free hyperparameter, but this is beyond the main thrust of this paper.

The loss function in (6) is unbounded. So, optimizing (6) without any constraints will also enlarge the within-class variance, which is a side

effect. To prevent this side effect from happening, we substitute (6) into (2), which gives the following optimization problem:

$$\mathbf{A}^* = \arg\min_{\mathbf{A}} f^{\mathrm{m}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) \tag{7}$$

where

$$f^{\mathrm{m}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) = -\sum_{i\in\mathrm{pos}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) + \alpha \sum_{i\in\mathrm{neg}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) + \lambda\|\mathbf{A}-\mathbf{A}_0\|_2^2 \tag{8}$$

and the superscript m denotes m-CML.

3.2.2. v-CML: an objective for reducing within-class variance

Minimizing the within-class variances of the decision scores of the true and imposter trials can be formulated as minimizing the following function:

$$\ell = \frac{1}{|\mathrm{pos}|-1}\sum_{i\in\mathrm{pos}}[S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) - \overline{S}_{\mathrm{pos}}]^2 + \frac{1}{|\mathrm{neg}|-1}\sum_{i\in\mathrm{neg}}[S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) - \overline{S}_{\mathrm{neg}}]^2 \tag{9}$$

where:

$$\overline{S}_{\mathrm{pos}} = \frac{1}{|\mathrm{pos}|}\sum_{i\in\mathrm{pos}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) \tag{10}$$

$$\overline{S}_{\mathrm{neg}} = \frac{1}{|\mathrm{neg}|}\sum_{i\in\mathrm{neg}} S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) \tag{11}$$

To constrain the loss function from its side effect which is the reduction of the between-class distance, we substitute (9) to (2) and get a new optimization problem, named v-CML:

$$\mathbf{A}^* = \arg\min_{\mathbf{A}} f^{\mathrm{v}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) \tag{12}$$

where

$$f^{\mathrm{v}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) = \sum_{i\in\mathrm{pos}}[S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) - \overline{S}_{\mathrm{pos}}]^2 + \alpha\sum_{i\in\mathrm{neg}}[S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) - \overline{S}_{\mathrm{neg}}]^2 + \lambda\|\mathbf{A}-\mathbf{A}_0\|^2 \tag{13}$$

and $\alpha = \frac{|\mathrm{pos}|-1}{|\mathrm{neg}|-1}$.

3.2.3. Initialization compensation

Many useful linear transformation can be used to initialize the matrix $\mathbf{A}_0$ in (4). In this work, we adopt LDA, WCCN or NAP, which are briefly described as follows.

LDA attempts to find a projection matrix $\mathbf{A}_0$ to minimize the intra-class variance caused by channel effects and meanwhile maximizes the variance between different speakers. This can be formulated as the following generalized eigenvalue decomposition problem:

$$\mathbf{S}_b\mathbf{V} = \mathbf{\Lambda}\mathbf{S}_w\mathbf{V} \tag{14}$$

where $\mathbf{S}_b$ and $\mathbf{S}_w$ are between- and within-class covariance matrices (Dehak et al., 2011), and $\mathbf{\Lambda}$ is a diagonal matrix consisting of eigenvalues. The projection matrix $\mathbf{A}_0$ is composed by the best eigenvectors (those with highest eigenvalues) of $\mathbf{V}$.

WCCN minimizes the error rate of false acceptance and false rejection Hatch et al. (2006). It can be explored through the inverse of the within class covariance matrix $\mathbf{W}$:

$$\mathbf{W} = \frac{1}{S}\sum_{s=1}^{S}\frac{1}{n_s}\sum_{i=1}^{n_s}(\boldsymbol{\omega}_i^s - \overline{\boldsymbol{\omega}}_s)(\boldsymbol{\omega}_i^s - \overline{\boldsymbol{\omega}}_s)^T \tag{15}$$

where $\overline{\omega}_s$ is the mean of the i-vectors/d-vectors of the s-th speaker. The compensation matrix $\mathbf{A}_0$ is then the Cholesky decomposition of $\mathbf{W}^{-1}$, i.e.,

$$\mathbf{W}^{-1} = \mathbf{A}_0^T\mathbf{A}_0 \tag{16}$$

NAP attempts to learn a linear transformation:

$$\mathbf{A}_0 = \mathbf{I} - \mathbf{R}\mathbf{R}^T \tag{17}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{R}$ is a low rank matrix, consisting of k eigenvectors of $\mathbf{W}$ in (15), which correspond to the k largest eigenvalues of $\mathbf{W}$.

3.3. Optimization algorithm

We use the L-BFGS algorithm (Liu and Nocedal, 1989)[1] to solve the optimization problems in (7) and (12). L-BFGS is a fast gradient descent algorithm, which enhances the robustness of our algorithms as it does not need to specify the learning rate manually. But in order to use the L-BFGS solver, we need to know the gradients of the optimization problems in (7) and (12), which are as follows:

3.3.1. Gradient of m-CML

The gradient of $f^{\mathrm{m}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i)$ in (7) with respect to $\mathbf{A}$ can be derived as:

$$\nabla f^{\mathrm{m}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) = -\sum_{i\in\mathrm{pos}}\frac{\partial S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A})}{\partial \mathbf{A}} + \alpha\sum_{i\in\mathrm{neg}}\frac{\partial S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A})}{\partial \mathbf{A}} + 2\lambda(\mathbf{A}-\mathbf{A}_0) \tag{18}$$

where

$$\frac{\partial S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A})}{\partial \mathbf{A}} = \frac{\partial\left\{\frac{\mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{y}_i}{\sqrt{\mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{x}_i}\sqrt{\mathbf{y}_i^T\mathbf{A}^T\mathbf{A}\mathbf{y}_i}}\right\}}{\partial \mathbf{A}} \tag{19}$$

For simplicity and clarity, we denote, respectively, the numerator and denominator of (19) by $u(\mathbf{A}) = \mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{y}_i$ and $v(\mathbf{A}) = \sqrt{\mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{x}_i}\sqrt{\mathbf{y}_i^T\mathbf{A}^T\mathbf{A}\mathbf{y}_i}$. It follows then that:

$$\nabla\left(\frac{u(\mathbf{A})}{v(\mathbf{A})}\right) = \frac{1}{v(\mathbf{A})}\frac{\partial u(\mathbf{A})}{\partial \mathbf{A}} - \frac{u(\mathbf{A})}{v(\mathbf{A})^2}\frac{\partial v(\mathbf{A})}{\partial \mathbf{A}} \tag{20}$$

$$\frac{\partial u(\mathbf{A})}{\partial \mathbf{A}} = \mathbf{A}\left(\mathbf{x}_i\mathbf{y}_i^T + \mathbf{y}_i\mathbf{x}_i^T\right) \tag{21}$$

$$\frac{\partial v(\mathbf{A})}{\partial \mathbf{A}} = \frac{\sqrt{\mathbf{y}_i^T\mathbf{A}^T\mathbf{A}\mathbf{y}_i}}{\sqrt{\mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{x}_i}}\mathbf{A}\mathbf{x}_i\mathbf{x}_i^T + \frac{\sqrt{\mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{x}_i}}{\sqrt{\mathbf{y}_i^T\mathbf{A}^T\mathbf{A}\mathbf{y}_i}}\mathbf{A}\mathbf{y}_i\mathbf{y}_i^T \tag{22}$$

Substituting (19)–(22) to (18) gives the final form of the gradient of $f^{\mathrm{m}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i)$.

3.3.2. Gradient of v-CML

Similarly, the gradient of $f^{\mathrm{v}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i)$ is:

$$\begin{aligned}\nabla f^{\mathrm{v}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i) &= \sum_{i\in\mathrm{pos}} 2\left[S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) - \frac{1}{|\mathrm{pos}|}\sum_{j\in\mathrm{pos}} S(\mathbf{x}_j, \mathbf{y}_j; \mathbf{A})\right] \\ &\times \left[\frac{\partial S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A})}{\partial \mathbf{A}} - \frac{1}{|\mathrm{pos}|}\sum_{j\in\mathrm{pos}}\frac{\partial S(\mathbf{x}_j, \mathbf{y}_j; \mathbf{A})}{\partial \mathbf{A}}\right] \\ &+ \alpha\sum_{i\in\mathrm{neg}} 2\left[S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A}) - \frac{1}{|\mathrm{neg}|}\sum_{j\in\mathrm{neg}} S(\mathbf{x}_j, \mathbf{y}_j; \mathbf{A})\right] \\ &\times \left[\frac{\partial S(\mathbf{x}_i, \mathbf{y}_i; \mathbf{A})}{\partial \mathbf{A}} - \frac{1}{|\mathrm{neg}|}\sum_{j\in\mathrm{neg}}\frac{\partial S(\mathbf{x}_j, \mathbf{y}_j; \mathbf{A})}{\partial \mathbf{A}}\right] \\ &+ 2\lambda(\mathbf{A}-\mathbf{A}_0)\end{aligned} \tag{23}$$

Using the results in (19)–(22), one can readily derive the final form of the gradient of $f^{\mathrm{v}}_{\mathbf{A}}(\mathbf{x}_i, \mathbf{y}_i, l_i)$ in (23).

[1] An efficient implementation of the L-BFGS algorithm can be found at Schmidt (2005).

Table 1
Test conditions. "EN" denotes the number of enrollment speakers, "TN" denotes the number of test speech segments, "Trials-N" denotes the number of trials, and C*i* denotes a test condition with *i* being the number of the enrollment speech segments provided from a speaker which varies from 1 to 5. The letters "K" and "M" after numbers are short for measurement "thousand" and "million" respectively.

Condition	Female			Male		
Name	EN	TN	Trials-N	EN	TN	Trial-N
C1	394	2748	1.08M	238	1650	393K
C2	390	2748	1.07M	236	1650	389K
C3	381	2748	1.05M	231	1650	381K
C4	362	2748	995K	221	1650	365K
C5	320	2748	879K	201	1650	332K

4. Experiments

In this section, we present some experimental results to validate the developed algorithms. Specifically, we evaluate the m-CML with the i-vector front-end as well as the v-CML with the d-vector front-end on the 2006 and 2008 NIST SRE corpora in Sections 4.1 to 4.5. Then, we compare m-CML and v-CML given both the x-vector and i-vector front-ends on the speaker in the wild (SITW) corpus in Section 4.6.

4.1. NIST SRE dataset

The first experiment was conducted on the 8 conversation conditions (8*conv* condition) of the 2006 and 2008 NIST SRE corpora. The 8*conv* condition of 2006 NIST SRE contains 402 female speakers and 298 male speakers. The 8*conv* condition of 2008 NIST SRE contains 395 female speakers and 240 male speakers. Each speaker has 8 conversations. A speaker utterance in a conversation was approximately 1 to 2 minutes long after removing the silence segments by voice activity detection (VAD), where the VAD label is determined by the ASR transcript. We split all speech signals into segments with a segment length of 15 seconds.

The 8*conv* condition of 2006 NIST SRE was used for development. It was divided into a total of 24043 segments for the female speakers and 17765 segments for the male speakers, which were used to train all models including the GMM-UBM/i-vector and d-vector front-ends, as well as the LDA, WCCN, NAP, and PLDA back-ends.

The 8 *conv* condition of 2008 NIST SRE was used for enrollment and test. In the enrollment stage, we selected 1 to 5 segments from the first conversation of a speaker as his/her enrollment data, which corresponds to 1 to 5 identity feature vectors after processing by a front-end. We took the mean of the identity feature vectors as the speaker model. In the test stage, we selected 1 segment from each of the remaining 7 conversations of the speaker for test, which corresponds to 7 test identity feature vectors. We took each speaker as a claimant with the remaining speakers acting as imposters, and rotated through the tests for all the speakers. We conducted the experiments on the female and male speakers separately. The number of trials are summarized in Table 1.

4.2. Experimental Setup on NIST SRE

The frame length and frame shift were set, respectively, to 25 and 10 ms. The 19 Mel frequency cepstral coefficients (MFCCs), 13 relative spectral filtered perceptual linear predictive cepstral coefficients (RASTA-PLP) and log energy, as well as their delta and double delta are used as the acoustic features, resulting a feature vector of 99 dimensions per frame (Chang and Wang, 2017).

4.2.1. Front-ends

The GMM-UBM/i-vector and d-vector were used as two front-ends. For the GMM-UBM/i-vector front-end, we employed the MSR Identity Toolbox (Sadjadi et al., 2013) as its implementation. Feature warping with a window size of 3 seconds was applied after the acoustic feature extraction. The number of Gaussian components of the gender-dependent GMM-UBM was set to 2048 (Chang and Wang, 2017). The dimension of the total variability matrix was set to 400.

For the d-vector front-end, the global cepstral mean and variance normalization was applied after the acoustic feature extraction. Gender-dependent feedforward neural networks were used as the DNN models. Each DNN model consists of 4 hidden layers with 512 hidden units per layer. We expanded each frame with a context window of 51 frames that centered at the current frame. The output dimension of the DNN was set to 402 for the female speakers, and 298 for the male speakers, which is the same as the number of the speakers in the development set. The d-vector of a speech segment is the average of the activations of the speech frames derived from the last DNN hidden layer.

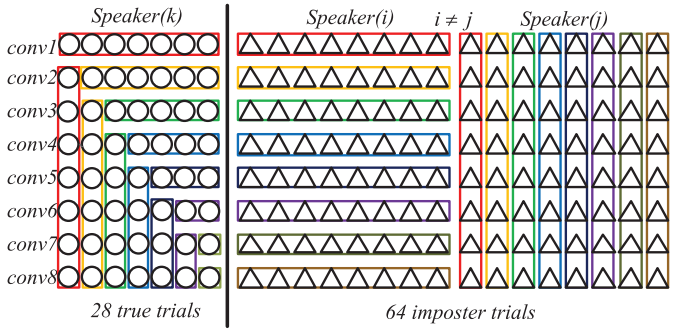

Fig. 4. Diagram for constructing training trials. Each circle or triangle stands for an identity feature vector extracted from a speech segment by a front-end.

4.2.2. Back-ends

For the m-CML and v-CML back-ends, we first extracted identity feature vectors from a front-end, and then constructed the training trials from the identity feature vectors. It is costly to use all pairs of speech segments for training, since the number of training trials in this case $N(N-1)/2$ where N is 24043 for the female speakers and 17765 for the male speakers. Here we developed a sampling strategy to balance the training accuracy and computational complexity as shown in Fig. 4. Specifically, to simulate the real-world test environments that the enrollment and test speech signals of a test trial may have different types of channel noise, we randomly selected 7 identity feature vectors from each conversation and combined them with the 7 identity feature vectors that came from the other 7 conversations respectively, which corresponds to 28 true trials. The construction of the true trials is as follows. As shown in the left-hand side of Fig. 4, each circle represents an identity vector. We first selected 7 identity vectors from each conversation where the identity vectors in the same row belong to the same conversation. Eight conversations of the speaker amount to 56 identity vectors. Then, the identity vectors in the box of the same color are grouped into non-overlapping trials. For example, we can obtain 7 trials from the red boxes by combining the 7 identity vectors in the red horizontal box with the 7 identity vectors in the red vertical box in sequence. The above example is repeated to other boxes. Finally, we obtain $7+6+\ldots+1=28$ true trials for each speaker. Similarly, we randomly selected 8 identity feature vectors from each conversation of a speaker and combined them with the 8 identity feature vectors that came from the other 8 conversations of any other speaker respectively, which corresponds to 64 imposter trials. As a result, the number of the training trials has been reduced to 5,169,720 for the female speakers and 2,840,536 for the male speakers. As will be analyzed in Section 4.5, we can further reduce the computational complexity dramatically by randomly down-sampling the imposter trials without suffering from a significant performance degradation. To summarize, in our experiments, the numbers of the training trials after down-sampling are listed in Table 2.

Table 2
Number of training trials.

	Female		Male	
	True trials	Imposter trials	True trials	Imposter trials
m-CML	10851	48289	7990	26220
v-CML	10865	48394	7990	26220

Table 3
EER comparison between m-CML and its initialization channel compensation techniques on the female speakers.

Method	C1	C2	C3	C4	C5
NULL	9.73%	6.63%	5.27%	4.62%	4.12%
LDA+PLDA	4.11%	3.76%	3.50%	3.50%	3.44%
LDA	6.88%	5.09%	4.37%	4.05%	3.74%
LDA+m-CML	4.58%	3.70%	3.37%	3.30%	3.08%
WCCN	5.34%	4.27%	3.74%	3.65%	3.37%
WCCN+m-CML	4.72%	3.97%	3.58%	3.52%	3.29%
NAP	5.57%	4.74%	4.35%	4.20%	3.94%
NAP+m-CML	4.87%	4.28%	3.98%	3.89%	3.67%

LDA, WCCN or NAP was used to initialize CML. We first tuned the hyperparameters of LDA and NAP by grid search from 50 to their maximum values with a step size of 50, and then selected their best hyperparameters in terms of EER. Note that the maximum output dimension of LDA can not be larger than $N-1$, where N is the number of the training speakers. Specifically, for the GMM-UBM/i-vector front-end, the output dimension of LDA was set to 200 for both the female and male speakers; the corank number of NAP was set to 350 for the female speakers and 100 for the male speakers. For the d-vector front-end, the output dimension of LDA was set to 400 for the female speakers and 297 for the male speakers; the corank number of NAP was set to 100 for both the female and male speakers. The m-CML with the three initialization techniques are denoted, respectively, as LDA + m-CML, WCCN + m-CML, and NAP + m-CML. The v-CML with the three initialization techniques are denoted, respectively, as LDA + v-CML, WCCN + v-CML, and NAP + v-CML.

The performance of the presented CML methods is evaluated in terms of EER and detection error tradeoff (DET) curve. For comparison, we also evaluate the performance of LDA, WCCN and NAP under the cosine similarity scoring framework. Also presented is the performance of the cosine similarity scoring back-end without any channel or session compensation techniques, denoted as "NULL", as well as the performance of the state-of-the-art LDA + PLDA back-end for comparison. The parameter settings of LDA and NAP are the same as those in the CML methods.

4.3. Results for m-CML with the i-vector front-end

In this subsection, GMM-UBM/i-vector is used as front-end for all studied methods.

4.3.1. Main results

Table 3 lists the EER results on the female speakers. From the table, one can see that the proposed CML methods outperform their initialization methods. Fig. 5 plots the relative performance improvement of the m-CML methods over their initialization methods. As seen, LDA + m-CML obtained an EER of relatively 33.4% lower than LDA in the C1 condition, and more than 15% in the C1 to C5 conditions. WCCN + CML and NAP + CML slightly outperform, respectively, WCCN and NAP.

Table 4 lists the EER results on the male speakers. Fig. 6 shows the relative performance improvement of the m-CML methods over their initialization methods. From the table and the figure, one can see that the experimental conclusion is similar with that on the female speakers. For example, LDA + m-CML achieves 16.3% relative improvement over LDA, and NAP + m-CML achieves 20.4% relative improvement over NAP in the C1 condition.

Table 4
EER comparison between m-CML and its initialization channel compensation techniques on the male speakers.

Method	C1	C2	C3	C4	C5
NULL	7.72%	5.90%	5.07%	4.61%	4.63%
LDA+PLDA	5.29%	4.90%	4.72%	4.40%	4.40%
LDA	6.62%	5.51%	4.89%	4.49%	4.55%
LDA+m-CML	5.54%	4.85%	4.47%	4.16%	4.20%
WCCN	5.71%	4.92%	4.56%	4.24%	4.28%
WCCN+m-CML	5.46%	4.83%	4.51%	4.20%	4.26%
NAP	7.16%	5.74%	4.95%	4.54%	4.59%
NAP+m-CML	5.70%	4.88%	4.42%	4.15%	4.19%

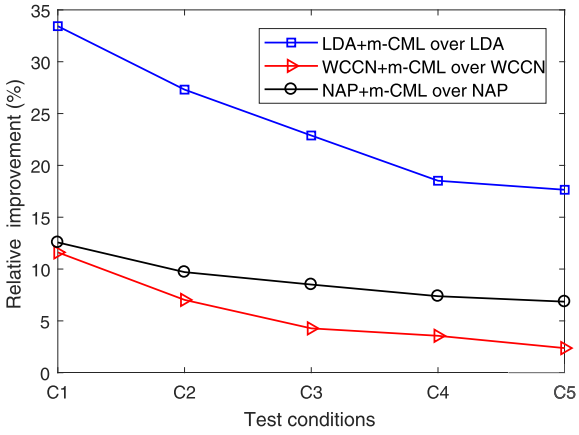

Fig. 5. Relative performance improvement of LDA + m-CML over LDA, WCCN + m-CML over WCCN, and NAP + m-CML over NAP respectively on the female speakers.

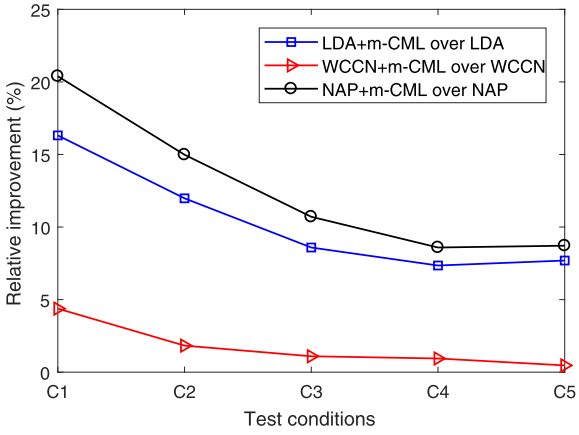

Fig. 6. Relative EER improvement of LDA + m-CML over LDA, WCCN + m-CML over WCCN, and NAP + m-CML over NAP respectively on the male speakers.

The performance of the state-of-the-art LDA + PLDA back-end is also presented in Tables 3 and 4 as a reference. Note that it is not fair or proper to compare m-CML with the LDA + PLDA back-end directly since m-CML aims to improve the performance of existing channel or session compensation methods under the cosine similarity scoring framework. The reasons that we focus on applying CML to the cosine similarity scoring without considering LDA + PLDA as a baseline are multiple, including but not limited to: 1) the cosine similarity scoring is not simpler than PLDA; 2) it does not rely on any model assumption and hence it can be improved, e.g., with the use of some deep embedding based front-ends (Garcia-Romero et al., 2019; Xie et al., 2019).

As seen, LDA + PLDA achieved EERs of 4.11% to 3.44% for the female speakers, and 5.29% to 4.40% from the C1 to C5 conditions for the

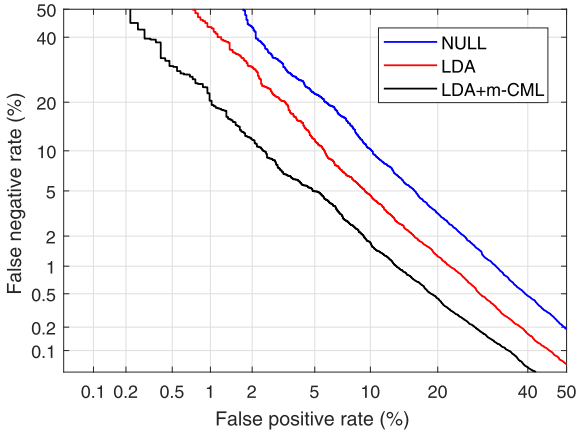

Fig. 7. DET curves produced by LDA, LDA + m-CML, and NULL in the C1 condition on the females.

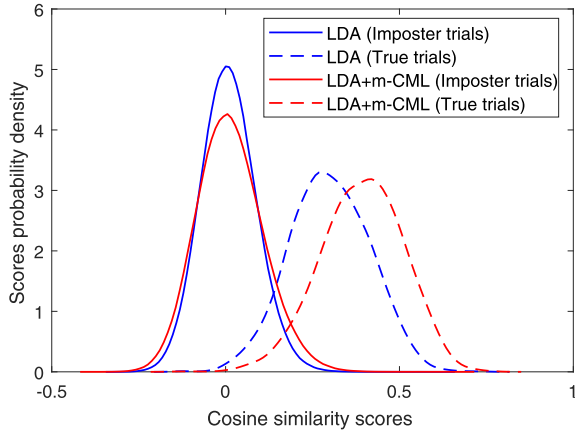

Fig. 8. Score distributions of LDA and LDA + m-CML in the C1 condition on the females.

male speakers. m-CML obtained better performance than LDA + PDLA in the C2 to C5 conditions. For example, LDA + m-CML achieved approximately 10.5% relative improvement in the C5 condition on the female speakers.

The DET curves of both LDA + m-CML and LDA in the female C1 condition are plotted in Fig. 7. One can see that LDA + m-CML yields a better DET performance than LDA.

4.3.2. *Effects of hyperparameter λ on performance*

Because the effects of λ on both genders are similar, we show only the results for the female speakers. Fig. 8 plots the decision score distributions of LDA and LDA + m-CML. From the figure, one can make the following observations. First, LDA + m-CML yielded a larger between-class distance than LDA. Second, although it produced a larger within-class variance of the decision scores of the imposter trials than LDA, LDA + m-CML yielded a smaller overlap region than LDA. As a result, LDA + m-CML has a smaller EER than LDA, which is justifies the motivation of this work as well as the feasibility of the presented methods.

Fig. 9 plots the EER as a function of the hyperparameter λ in the C1 condition. It is seen that the EER curve of LDA + m-CML first decreases and then increases along with the value of λ. The underlying reason can be explained as follows. If the value of λ is small, the regularization term in (4) does not play an important role. Consequently, CML does not only increase the between-class distance, but also the within-class variance as a side effect. As a result, the EER performance is not improved compared to its initialization. At the extreme case when the value of λ is close to zero, the regularization term may be neglected, leading to performance

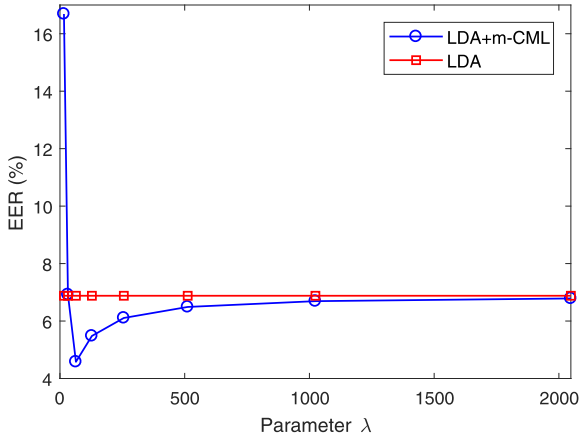

Fig. 9. EER of LDA + m-CML and LDA as a function of the hyperparameter λ in the C1 condition for the female speakers.

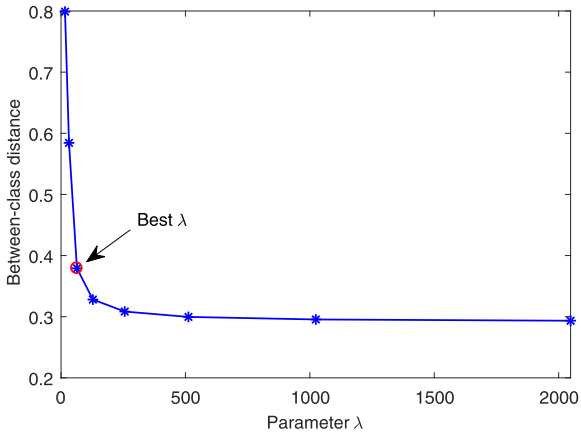

Fig. 10. Between-class distance of LDA + m-CML as a function of the hyperparameter λ in the C1 condition for the female speakers.

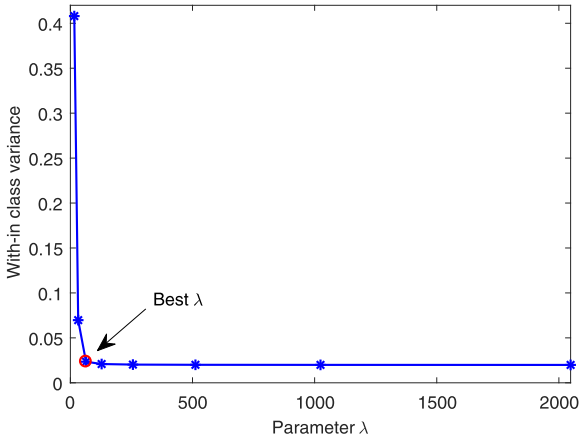

Fig. 11. Within-class variance of LDA + m-CML as a function of the hyperparameter λ in the C1 condition for the female speakers.

degradation. On the other hand, if the value of λ is too large, CML approaches to its initialization point. The same experimental phenomenon was observed for WCCN + CML and NAP + CML.

The aforementioned experimental phenomena can be further explained by analyzing the results in Figs. 10 and 11 jointly, where the two figures show how the between-class distance and within-class variance vary along with the value of λ, respectively. As seen, when the

Table 5
EER comparison between v-CML and its initialization channel compensation techniques on the female speakers.

Method	C1	C2	C3	C4	C5
NULL	13.24%	12.90%	12.51%	12.48%	12.33%
LDA+PLDA	8.27%	7.74%	7.39%	7.38%	7.39%
LDA	8.93%	8.47%	8.14%	8.16%	8.13%
LDA+v-CML	7.96%	7.52%	7.18%	7.20%	7.12%
WCCN	10.06%	9.52%	9.12 %	9.05%	8.90%
WCCN+v-CML	8.47%	8.03%	7.63%	7.67%	7.44%
NAP	10.87%	9.86%	9.33%	9.10%	8.97%
NAP+v-CML	8.77%	8.15%	7.72%	7.68%	7.48%

Table 6
EER comparison between v-CML and its initialization channel compensation techniques on the male speakers.

Method	C1	C2	C3	C4	C5
NULL	14.22%	14.01%	13.58%	13.33%	13.34%
LDA+PLDA	10.14%	9.75%	9.28%	8.81%	8.96%
LDA	11.18%	10.84%	10.46%	10.01%	10.08%
LDA+v-CML	9.93%	9.48%	9.06%	8.55%	8.61%
WCCN	11.33%	10.97%	10.58%	10.18%	10.21%
WCCN+v-CML	9.63%	9.34%	8.91%	8.49%	8.44%
NAP	11.88%	11.11%	10.58%	9.99%	10.02%
NAP+v-CML	10.11 %	9.55%	9.20%	8.67%	8.72%

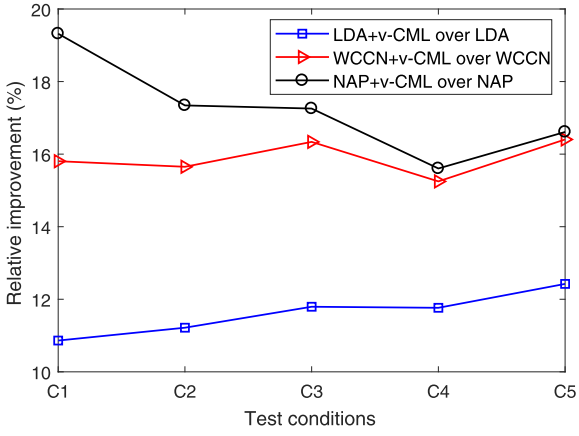

Fig. 12. Relative EER improvement of LDA + v-CML over LDA, WCCN + v-CML over WCCN and NAP + v-CML over NAP on the females.

value of the hyperparameter λ varies from infinity to zero, the between-class distance and within-class variance are both becoming larger. However, enlarging the within-class variance is a side effect, which should be avoided through setting the proper value of λ. As seen from the two figures, the value of λ that gives the lowest EER is the point where the between-class distance has been enlarged significantly while the within-class variance begins to increase quickly.

4.4. Results of v-CML with the d-vector front-end

In this subsection, d-vector is used as the front-end for all the studied methods.

Tables 5 and 6 list the results of v-CML on the female and male speakers, respectively. Figs. 12 and 13 plot the relative performance improvement of v-CML over different initialization techniques, respectively. From the results, we observe the effectiveness of v-CML. For example, LDA + v-CML obtained 0.97% absolute EER improvement over LDA in the C1 condition, and more than 10% relative EER improvement in the C2 to C5 conditions for the female speakers. Fig. 14 plots the DET curves of LDA, LDA + v-CML, and NULL in the C1 condition for the female speakers. As seen, LDA + CML yielded a better DET performance

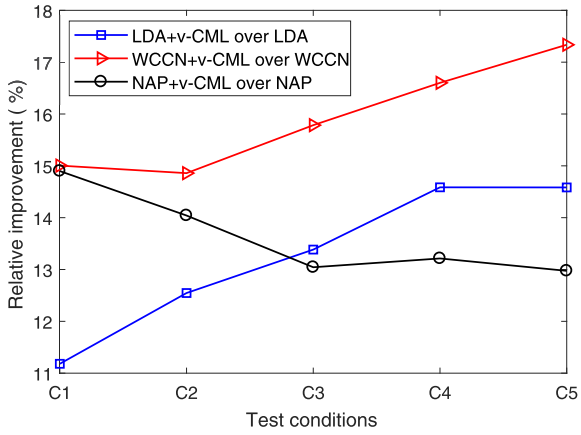

Fig. 13. Relative EER improvement of LDA + v-CML over LDA, WCCN + v-CML over WCCN and NAP + v-CML over NAP on the males.

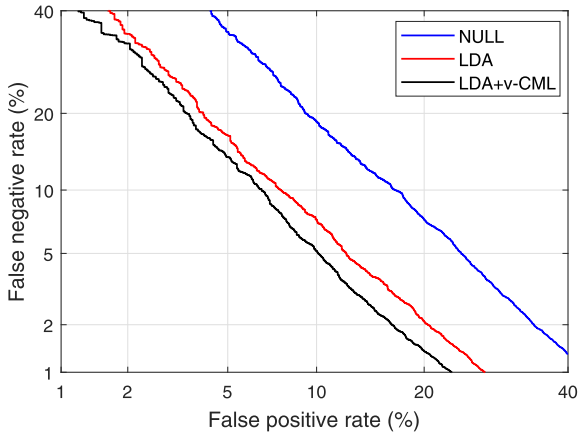

Fig. 14. DET curves produced by LDA, LDA + v-CML and NULL in the C1 condition on the females.

than both LDA and NULL. The performance of PLDA is also shown as a reference. PLDA achieves EERs of 8.27% to 7.39% in the five test conditions for the females, and 10.14% to 8.96% for the male speakers. It is seen that LDA + v-CML achieved better EER performance than PLDA in the female dataset.

Note that one can also see from the tables and figures that the performance of the v-CML with the d-vector front-end is worse than the m-CML with the i-vector front-end. The reasons that the DNN based d-vector system does not outperform the traditional GMM based i-vector system have been studied extensively. One of the core issues is that the d-vector front-end does speaker classification at the frame level, which may not be reliable as a single frame has too little statistic information about the speaker. We should point out that that the inferior performance of the d-vector front-end to the i-vector front-end does not affect the validation of the effectiveness of v-CML.

To study the effect of hyperparameter λ on the EER performance, we carried out similar experiments with Section 4.3.2 for LDA + v-CML. Fig. 15 plots the decision score distributions of LDA and LDA + v-CML in the C5 condition on the females, and Fig. 16 plots the decision score distributions of NAP and NAP + v-CML in the C1 condition on the females. From these two figures, one can see that v-CML yields a smaller within-class variance than LDA and NAP while keeping a similar between-class distance with the latter, which results in a smaller overlap region than LDA and NAP. Fig. 17 is the EER performance of LDA + v-CML with respect to different λ in the C1 condition. From the figure, one can see that the EER of LDA + v-CML has the similar variation tendency with LDA + m-CML with respect to λ.

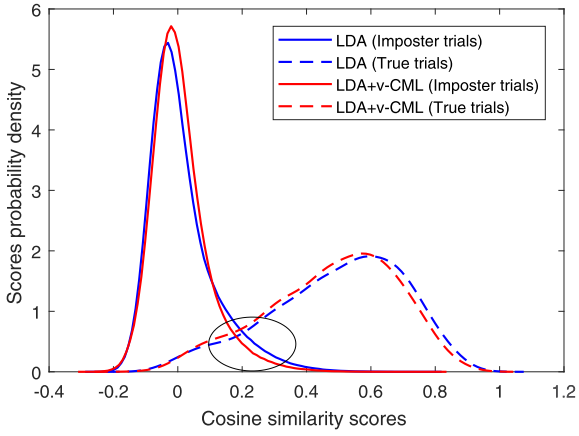

Fig. 15. Score distributions of LDA and LDA + v-CML in the C5 condition on the females.

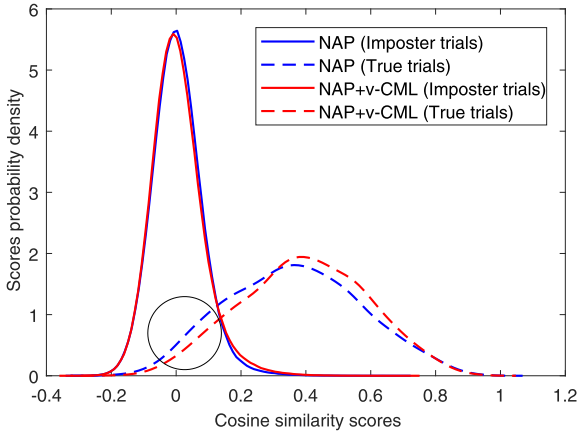

Fig. 16. Score distributions of NAP and NAP + v-CML in the C1 condition on the females.

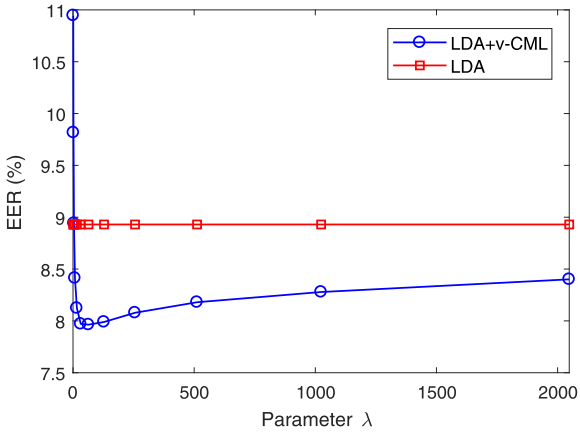

Fig. 17. EER of LDA + v-CML with respect to hyperparameter λ in the C1 condition on the female speakers.

4.5. Analysis of computational complexity

To evaluate the relationship between the test accuracy and training time complexity, we fixed the number of the true training trials to 10.8 thousand, and varied the number of the imposter training trials from 4.8 million to 48 thousand by down-sampling on the females as shown in Table 7. From the table, we observe that the EER of CML does not drop significantly by down-sampling, while the training time can be greatly reduced. For clearly, we provide two examples in Fig. 1.

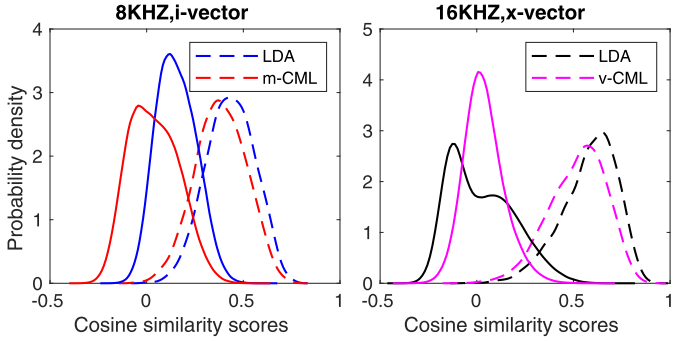

Fig. 18. Score distributions produced by different back-ends. The terms "m-CML" and "v-CML" are short for "LDA + m-CML" and "LDA + v-CML" respectively.

Table 7
Relationship between test accuracy and training time on the females. The letters "K" and "M" after numbers are short for measurement "thousand" and "million" respectively.

Number of imposter trials		48K	241K	482K	2.4M	4.8M
m-CML	EER (%)	4.56	4.54	4.53	4.52	4.49
	Time (in seconds)	35	95	205	762	1445
v-CML	EER (%)	8.06	8.05	7.94	8.03	8.03
	Time (in seconds)	247	982	1747	7950	16410

Table 8
Comparison results of CML with different front-ends on the 8-kHz and 16-kHz systems respectively.

System	Front-ends	i-vector		x-vector	
	Method	EER(%)	DCF10^{-2}	EER(%)	DCF10^{-2}
8KHZ	LDA	11.86	0.684	8.53	0.600
	LDA+m-CML	10.02	0.708	8.33	0.598
	LDA+v-CML	10.55	0.682	7.00	0.552
16KHZ	LDA	7.45	0.606	5.41	0.465
	LDA+m-CML	7.45	0.607	5.41	0.465
	LDA+v-CML	6.57	0.517	3.95	0.380

4.6. Experiments of CML with the x-vector front-end

It is known that the x-vector front-end, as a DNN based method, has achieved better performance than the i-vector and d-vector front-ends in many cases. Here we study the CML with the x-vector front-end, and take the CML with the i-vector front-end as a reference. For simplicity, we employed the development core trials of SITW to train CML, and tested its effectiveness on the evaluation core trials of SITW. All other parts of the comparison systems followed the Kaldi (Povey et al., 2011) recipes of "*/kaldi-master/egs/sre16/*" and "*/kaldi-master/egs/sitw/*". We name the two recipes as the *8-kHz system* and *16-kHz system* respectively for simplicity.

Table 8 lists the comparison results in terms of EER and the minimum detection cost function with $P_{target} = 10^{-2}$ (DCF10^{-2}). We see from the table that, for the 8-kHz system, when the i-vector front-end is used, m-CML achieves better performance than v-CML; when the x-vector front-end is applied, the performance of v-CML is significantly better than that of m-CML. The underlying reason is that the similarity scores of the x-vectors are heavy tailed, while the similarity scores of the i-vectors follow a Gaussian distribution. For the 16-kHz system, v-CML always obtains better performance than m-CML. We also see that m-CML is ineffective in this evaluation, due to the phenomenon that the similarity

scores of both the i-vectors and x-vectors are heavy tailed. To support our above analysis, we show two examples of the score distributions produced from the 8-kHz i-vector system and 16-kHz x-vector system respectively in Fig. 18. As a byproduct, it is clear that the CMLs with the x-vector front-end perform better than their counterparts with the i-vector front-end, which agrees with the recent research observations on speaker verification.

5. Conclusions and future work

In this paper, we presented two back-end metric learning methods, named m-CML and v-CML, to improve the EER performance of speaker verification. m-CML aims to increase the between-class distance of the decision scores of true and imposter trials while keeping the within-class variance unchanged by a regularization term. In comparison, v-CML attempts to minimize the within-class variance of the decision scores while maintaining the between-class distance unchanged by a similar regularization term. The regularization terms are initialized by traditional channel or session compensation techniques. In evaluation, we combined m-CML with the GMM-UBM/i-vector front-end, and combined v-CML with the d-vector or x-vector front-end. Experimental results on the NIST SRE and SITW datasets demonstrated that the two CML methods outperform their initialization methods, and their performance is comparable to that of the state-of-the-art PLDA back-end.

Our work in progress is to find a better initialization compensation technique so as to further improve the performance of CML in the system level, given the fact that CML is lower bounded by its initialization transformation matrix $\mathbf{A}_0$. We will also study the generalization ability of the algorithm in the future.

Declaration of Competing Interest

We declare no conflict of interests.

CRediT authorship contribution statement

Zhongxin Bai: Conceptualization, Methodology, Formal analysis, Software, Writing - original draft. **Xiao-Lei Zhang:** Methodology, Resources, Data curation, Writing - review & editing, Project administration. **Jingdong Chen:** Visualization, Investigation, Writing - review & editing, Supervision, Funding acquisition.

References

Ahmad, W., Karnick, H., Hegde, R.M., 2014. Cosine distance metric learning for speaker verification using large margin nearest neighbor method. In: Pacific Rim Conference on Multimedia. Springer, pp. 294–303.

Bai, Z., Zhang, X.-L., Chen, J., 2018. Cosine metric learning for speaker verification in the i-vector space. In: Proc. Interspeech 2018, pp. 1126–1130.

Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.

Campbell, W.M., 2014. Using deep belief networks for vector-based speaker recognition. In: Fifteenth Annual Conference of the International Speech Communication Association.

Campbell, W.M., Sturim, D.E., Reynolds, D.A., 2006. Support vector machines using GMM supervectors for speaker verification. IEEE sS Process. Lett. 13 (5), 308–311.

Campbell, W.M., Sturim, D.E., Reynolds, D.A., Solomonoff, A., 2006. SVM based speaker verification using a gmm supervector kernel and nap variability compensation. In: Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on, 1. IEEE. I–I

Chang, J., Wang, D., 2017. Robust speaker recognition based on DNN/i-vectors and speech separation. In: Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on. IEEE, pp. 5415–5419.

Cumani, S., Laface, P., 2014. Large-scale training of pairwise support vector machines for speaker recognition. IEEE/ACM Trans. Audio SpeechLang. Process. 22 (11), 1590–1600.

Cumani, S., Laface, P., Cumani, S., Laface, P., 2017. Nonlinear i-vector transformations for PLDA-based speaker recognition. IEEE/ACM Trans. Audio Speech Lang.Process. (TASLP) 25 (4), 908–919.

Dehak, N., Kenny, P.J., Dehak, R., Dumouchel, P., Ouellet, P., 2011. Front-end factor analysis for speaker verification. IEEE Trans. Audio Speech Lang.Process. 19 (4), 788–798.

Fang, X., et al., 2013. Bayesian Distance Metric Learning on i-Vector for Speaker Verification. Massachusetts Institute of Technology.

Garcia-Romero, D., Espy-Wilson, C.Y., 2011. Analysis of i-vector length normalization in speaker recognition systems. In: Twelfth Annual Conference of the International Speech Communication Association.

Garcia-Romero, D., Snyder, D., Sell, G., McCree, A., Povey, D., Khudanpur, S., 2019. X-vector dnn refinement with full-length recordings for speaker recognition. In: Proc. Interspeech, pp. 1493–1496.

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

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

Hasan, T., Hansen, J.H., 2014. Maximum likelihood acoustic factor analysis models for robust speaker verification in noise. IEEE/ACM Trans. Audio Speech Lang.Process. (TASLP) 22 (2), 381–391.

Hatch, A.O., Kajarekar, S., Stolcke, A., 2006. Within-class covariance normalization for SVM-based speaker recognition. Ninth international conference on spoken language processing.

Heigold, G., Moreno, I., Bengio, S., Shazeer, N., 2016. End-to-end text-dependent speaker verification. In: Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on. IEEE, pp. 5115–5119.

Kenny, P., 2010. Bayesian speaker verification with heavy-tailed priors. In: Odyssey, p. 14.

Kenny, P., Gupta, V., Stafylakis, T., Ouellet, P., Alam, J., 2014. Deep neural networks for extracting baum-welch statistics for speaker recognition. In: Proc. Odyssey, pp. 293–298.

Kinnunen, T., Li, H., 2010. An overview of text-independent speaker recognition: From features to supervectors. Speech Commun. 52 (1), 12–40.

Lei, Y., Scheffer, N., Ferrer, L., McLaren, M., 2014. A novel scheme for speaker recognition using a phonetically-aware deep neural network. In: Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on. IEEE, pp. 1695–1699.

Li, L., Wang, D., Xing, C., Zheng, T.F., 2016. Max-margin metric learning for speaker recognition. In: Chinese Spoken Language Processing (ISCSLP), 2016 10th International Symposium on. IEEE, pp. 1–4.

Li, N., Mak, M.-W., 2015. Snr-invariant PLDA modeling in nonparametric subspace for robust speaker verification. IEEE/ACM Trans. Audio Speech Lang. Process. (TASLP) 23 (10), 1648–1659.

Li, N., Mak, M.-W., Chien, J.-T., 2017. DNN-driven mixture of PLDA for robust speaker verification. IEEE/ACM Trans. Audio SpeechLang. Processi. 25 (6), 1371–1383.

Liu, D.C., Nocedal, J., 1989. On the limited memory BFGS method for large scale optimization. Math. Programm. 45 (1–3), 503–528.

Mak, M.-W., Pang, X., Chien, J.-T., 2016. Mixture of PLDA for noise robust i-vector speaker verification. IEEE/ACM Trans. Audio Speech Lang. Process. (TASLP) 24 (1), 130–142.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., et al., 2011. The kaldi speech recognition toolkit. IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. IEEE Signal Processing Society.

Prince, S.J., Elder, J.H., 2007. Probabilistic linear discriminant analysis for inferences about identity. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. IEEE, pp. 1–8.

Reynolds, D.A., Quatieri, T.F., Dunn, R.B., 2000. Speaker verification using adapted gaussian mixture models. Digital Signal Process. 10 (1–3), 19–41.

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

Richardson, F., Reynolds, D., Dehak, N., 2015b. A unified deep neural network for speaker and language recognition. arXiv:1504.00923.

Sadjadi, S.O., Slaney, M., Heck, L., 2013. MSR identity toolbox v1.0: A Matlab toolbox for speaker-recognition research. Speech Lang. Process. Tech. Comm. Newslett. 1 (4), 1–32.

Schmidt, M., 2005. minfunc: unconstrained differentiable multivariate optimization in Matlab. http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html.

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

Snyder, D., Garcia-Romero, D., Povey, D., Khudanpur, S., 2017. Deep neural network embeddings for text-independent speaker verification. In: Proc. Interspeech, pp. 999–1003.

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., Khudanpur, S., 2018. X-vectors: Robust dnn embeddings for speaker recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE.

Snyder, D., Ghahremani, P., Povey, D., Garcia-Romero, D., Carmiel, Y., Khudanpur, S., 2016. Deep neural network-based speaker embeddings for end-to-end speaker verification. In: Spoken Language Technology Workshop (SLT), 2016 IEEE. IEEE, pp. 165–170.

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

Tan, Z., Mak, M.-W., Mak, B.K.-W., 2018. DNN-based score calibration with multitask learning for noise robust speaker verification. IEEE/ACM Trans. Audio Speech Lang.Process. (TASLP) 26 (4), 700–712.

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

Wan, L., Wang, Q., Papir, A., Moreno, I. L., 2017. Generalized end-to-end loss for speaker verification. arXiv:1710.10467.

Wang, D., Li, L., Tang, Z., Zheng, T. F., 2017. Deep speaker verification: do we need end to end? arXiv:1706.07859.
Xie, W., Nagrani, A., Chung, J.S., Zisserman, A., 2019. Utterance-level aggregation for speaker recognition in the wild. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 5791–5795.
Yaman, S., Pelecanos, J., Sarikaya, R., 2012. Bottleneck features for speaker recognition. Odyssey 2012-The Speaker and Language Recognition Workshop.
Z.G., Tieran, H., Jiqing, 2018. Deep neural network based discriminative training for i-vector/PLDA speaker verification. In: Acoustics, Speech and Signal Processing (ICASSP), 2018 IEEE International Conference on. IEEE, pp. 5354–5358.