POLITECNICO DI TORINO
Repository ISTITUZIONALE

Analysis of back-end classifiers for speaker verification with DNN embeddings

Original
Analysis of back-end classifiers for speaker verification with DNN embeddings / Cumani, S., Barahona, S., Rohdin, J., Silnova, A., Plchot, O.. - In: COMPUTER SPEECH AND LANGUAGE. - ISSN 0885-2308. - 103:(2027). [10.1016/j.csl.2026.102059]

Availability:
This version is available at: 11583/3016443 since: 2026-10-08T15:01:02Z

Publisher:
Elsevier

Published
DOI:10.1016/j.csl.2026.102059

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

27 October 2026

Computer Speech and Language 103 (2027) 102059

Contents lists available at ScienceDirect

Computer Speech & Language

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

Analysis of back-end classifiers for speaker verification with DNN embeddings

Sandro Cumani [a,*], Sara Barahona [b], Johan Rohdin [c], Anna Silnova [c], Oldřich Plchot [c]

[a] Politecnico di Torino, Turin, Italy
[b] AUDIAS, Universidad Autónoma de Madrid, Madrid, Spain
[c] Brno University of Technology, Brno, Czechia

ARTICLE INFO

Keywords:
Speaker recognition
Speaker verification back-end
Pairwise Support Vector Machine
Toroidal Probabilistic Spherical Discriminant Analysis
Cosine scoring
Probabilistic Linear Discriminant Analysis

ABSTRACT

The rapid evolution of DNN-based speaker embeddings has brought significant improvements in the field of speaker recognition. The increased focus on DNN architectures, however, has been accompanied by a reduced interest and a significant simplification of the embedding classification stage, due in part to the poor performance of Probabilistic Linear Discriminant Analysis (PLDA) and in part to the success of cosine scoring for comparing speaker embeddings. As a result, PLDA has been gradually replaced by cosine scoring. In this work, we critically evaluate the effectiveness of cosine distance with respect to alternative generative and discriminative classification methods. We compare cosine distance with several alternative classifiers over a large set of publicly available benchmarks with different supervised DNN embedding extractors, showing that substantial gains can be obtained by more complex back-end models, whereas over-reliance on cosine scoring may, in some cases, even lead to misleading conclusions on the relative performance of different DNN architectures.

1. Introduction

DNN embeddings have become the standard approach for utterance representation in the field of speaker recognition. Embedding-based speaker verification systems employ a classification back-end that allows comparing a test embedding with one or more enrollment embeddings for the target speaker. Before the recent shift towards DNNs, significant effort had been devoted to developing classification back-ends, which proved critical for good performance. The success of neural embedding extractors, and in particular models trained with angular distances (Deng et al., 2019), has been accompanied by a decline of interest in back-end modeling. This can be partially explained by the poor performance of Probabilistic Linear Discriminant Analysis (PLDA) (Ioffe, 2006; Prince and Elder, 2007; Kenny, 2010), which had previously been the standard approach for i-vector (Dehak et al., 2011) comparison, and the success of cosine-based DNN training objectives, which resulted in the simple cosine scoring becoming competitive and even able to outperform PLDA (Wang et al., 2022). This, in turn, has led many authors to focus on the DNN architecture, while relying solely on cosine distance for utterance comparison (Desplanques et al., 2020; Thienpondt and Demuynck, 2023; Yakovlev et al., 2024). The pairwise cosine distance may seem a natural choice for DNNs trained with angular objective functions, since such objectives aim at increasing the angular distance between training utterances belonging to different speakers with respect to the angular distance of embeddings belonging to a single speaker. However, it should be noted that embedding extractors are typically trained to perform closed-set classification on the training set, and therefore, there is no guarantee that the comparison function employed by the network will generalize to utterances from unseen speakers. Furthermore, typical use cases involve scenarios whose

* Corresponding author.
E-mail address: sandro.cumani@polito.it (S. Cumani).

https://doi.org/10.1016/j.csl.2026.102059
Received 18 March 2026; Received in revised form 28 September 2026; Accepted 29 September 2026
Available online 5 October 2026
0885-2308/© 2026 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

acoustic characteristics may differ from those of the DNN training set. While fine-tuning mechanisms allow addressing domain mismatch, these involve partially re-training the embedding extractor, which may be expensive and not always viable.

Recent works have shown that proper back-end models can substantially improve the accuracy and robustness of speaker verification systems compared to cosine distance. In Silnova et al. (2023), the authors have proposed a new generative model, the Toroidal Probabilistic Spherical Discriminant Analysis (TPSDA), that accounts for the characteristics of DNN embeddings trained with angular losses. The proposed approach is able to characterize speaker similarity in terms of angular distances between embeddings, and can substantially outperform cosine. In Cumani et al. (2025), we have shown that discriminative models developed for i-vectors (Dehak et al., 2011), such as the Pairwise Support Vector Machine (PSVM) (Cumani et al., 2013; Cumani and Laface, 2014b), can also substantially outperform cosine scoring. In this work, we extend our previous analysis through a systematic benchmarking of classification back-ends over a wide range of evaluation sets with several different modern supervised front-end embedding extractors. Compared to previous studies, we aim at providing a systematic comparison of robust back-ends that, while already known to be effective in different scenarios, have not been directly compared in the literature. A second goal of our work is to analyze how well robust classifiers such as PSVM, developed for previous state-of-the-art utterance representations and rarely used in the context of DNN embeddings classification, behave with newer DNN-based utterance representations, in order to preserve methods that prove to be still effective. Finally, we also provide novel insights on possible risks due to over-reliance on cosine scoring to compare different front-end extractors. Our analysis focuses on five alternative back-ends, PLDA, the approximated Heavy-Tailed (HT-PLDA) of Silnova et al. (2018) and Brümmer et al. (2018), the TPSDA model of Silnova et al. (2023), the PSVM model of Cumani and Laface (2014b) and the discriminatively trained PLDA (DPLDA) model of Burget et al. (2011). These back-ends have been selected for their superior performance with previous state-of-the-art utterance representations (PLDA, HT-PLDA, PSVM, DPLDA), for their superior theoretical framework in modeling directional data (TPSDA) and for their effectiveness in adapting to different data distribution thanks to an adaptable scoring function and a robust discriminative training criterion (PSVM, DPLDA). We consider different supervised front-end embedding extractors, including both our own and off-the-shelf pre-trained models. We analyze the performance of the chosen back-ends over several standard benchmarks, including NIST SRE 2018 (NIST, 2018), 2019 (NIST, 2019), 2021 (NIST, 2021), and 2024 (NIST, 2024), CNCeleb (Li et al., 2023), SITW (McLaren et al., 2016) and VoxCeleb (Nagrani et al., 2019). We show that proper back-end modeling can substantially improve DNN embedding classification with respect to cosine scoring, both in matched and mismatched domain scenarios. Furthermore, robust back-ends can also leverage additional training data, further improving results in scenarios where fine-tuning the embedding extractor is not viable. Finally, our results show that over-reliance on cosine scoring may bias comparisons of DNN embedding extractors, leading, in some cases, to suboptimal conclusions about the relative performance of different architectures.

The rest of the paper is organized as follows. Section 2 describes the front-end embedding extractors. Section 3 recalls the main characteristics of the back-ends under analysis. In Section 4, we provide and analyze our experimental results. Conclusions are given in Section 5.

2. Front-end embedding extractors

In this section, we introduce the diverse set of front-ends used in our back-end analysis. We consider off-the-shelf, publicly available pre-trained extractors, and we also explore the effect of adapting pre-trained models on large-scale datasets with additional data, including conversational telephone speech (CTS), VoxCeleb2 (Chung et al., 2018), or a combination of both, using the WeSpeaker toolkit (Wang et al., 2024, 2023).[1] When adaptation is performed, it follows a large-margin fine-tuning strategy aimed at enhancing speaker discrimination, leveraging longer speech segments and increased angular margins when using the Additive Angular Margin (AAM) (Deng et al., 2019) loss. Table 1 summarizes the architectures and the individual models that were employed for each architecture, where VC2 and CTS + VC2 (Mix) correspond to different fine-tuning training lists, as detailed in Section 4.2.

2.1. ResNet

Residual Networks (ResNet) (He et al., 2016) have shown strong performance in speaker recognition across different evaluations (Sadjadi et al., 2022; Huh et al., 2024). ResNet backbones are typically trained as embedding extractors that map variable-length acoustic features into fixed-dimensional speaker embeddings. To achieve this, frame-level features from the final convolutional layer are aggregated using a pooling mechanism. While early models relied on simple Temporal Statistics Pooling (TSTP) (Snyder et al., 2018), which calculates the mean and standard deviation across time, more recent architectures frequently employ Attentive Statistics Pooling (ATP) (Okabe et al., 2018) to adaptively weight discriminative speech segments.

We evaluate different ResNet configurations to provide a diverse set of front-ends for our back-end analysis. We include a ResNet-34 pre-trained on VoxCeleb2, trained with both standard Softmax and AAM loss (Deng et al., 2019) to compare the effect on back-ends of margin-based discrimination. We also evaluate the impact of large-scale training by including larger ResNet variants pre-trained on VoxBlink2 (Lin et al., 2024) (over 110k speakers), which has been shown to substantially improve model generalization. Specifically, we incorporate a ResNet-152[2] from our NIST-SRE24 submission (Barahona et al., 2025), pre-trained on both original VoxBlink2 data and the same data down-sampled to 8 kHz and up-sampled again to 16 kHz after randomly applying

[1] https://github.com/wenet-e2e/wespeaker.
[2] https://huggingface.co/sarabarahona/voxblink2-ResNet152-16k.

Table 1
Summary of the analyzed DNN embedding extractors. Off-the-shelf refers to publicly available models. All other models have been trained from scratch or fine-tuned using the WeSpeaker toolkit, on either VoxCeleb2 (VC2) or VoxCeleb2 and NIST CTS Superset (Mix) data. The column "size" refers to the embedding dimension. The FTDNN number in parentheses refers to the embedding dimension after PCA (see Section 4.3 for details).

Architecture	Loss	Fine tuning	Off-the-shelf (Pre-trained)	Size	Architecture	Loss	Fine tuning	Off-the-shelf (Pre-trained)	Size
SAM-ResNet-100	AAM	None	✓	256	ECAPA	AAM	None	✓	192
		VC2	✓				VC2	✓	
		Mix	✗				Mix	✗	
ResNet-152	AAM	None	✗	256	FTDNN	AAM	None	✗	192 (180)
		VC2	✗				VC2	✗	
		Mix	✗				Mix	✗	
						Softmax	None	✗	
ResNet-34	AAM	None	✓	256	ReDimNet B0	AAM	VC2	✓	192
		VC2	✓		ReDimNet B3			✓	
		Mix	✗		ReDimNet B6			✓	
	Softmax	None	✗		ECAPA2	AAM	VC2	✓	192

the GSM codec to half of the segments, to simulate the telephone channel effect. Finally, we examine a Simple Attention Modules (SimAM) (Yang et al., 2021) ResNet-100 (SAM-ResNet-100 in the following) also pre-trained on VoxBlink2, which replaces TSTP with ATP and integrates Simple Attention Modules (Yang et al., 2021) within the residual blocks to extract more refined speaker features without increasing the total number of model parameters.

2.2. ECAPA-TDNN and ECAPA2

The ECAPA-TDNN (Desplanques et al., 2020) (simply ECAPA in the following) improves upon the standard Time Delay Neural Network (TDNN) by incorporating 1D-Res2Net modules (Gao et al., 2019) and Squeeze-and-Excitation (SE) blocks to capture multi-scale, channel-dependent features. Unlike ResNet-based models that apply 2D convolutions, ECAPA-TDNN processes information primarily along the temporal axis and aggregates features across all residual blocks. In our experiments, we use a 512-channel ECAPA-TDNN[3] pre-trained on VoxCeleb2.

We also evaluate ECAPA2 (Thienpondt and Demuynck, 2023), an extension of ECAPA that combines TDNN layers for explicit global temporal modeling with 2D convolutions to capture local time-frequency patterns. This architecture enriches local spectral representation while preserving long-term temporal context. For our analysis, we use an ECAPA2[4] model pre-trained on VoxCeleb2 to assess the impact of this 1D-2D design on back-end classifiers.

2.3. ReDimNet

Following hybrid network designs, we also evaluate Reshape Dimensions Network (ReDimNet) (Yakovlev et al., 2024), which integrates 1D and 2D convolutional blocks within a single architecture by dynamically reshaping feature map dimensions. This design enables the model to alternate between time–frequency and temporal representations, capturing complementary speech characteristics, and has been shown to achieve state-of-the-art performance on the VoxCeleb benchmark compared to ResNet-based models. We explore the scalability of this architecture by analyzing different sizes (B0, B3 and B6),[5] all pre-trained on the VoxCeleb2 dataset.

2.4. F-TDNN

The factorized TDNN (F-TDNN) architecture was introduced for speech recognition in Povey et al. (2018) and applied to speaker recognition in Villalba et al. (2020). It extends TDNN by replacing the weight matrix of each TDNN layer by a product of two matrices (factors) of lower rank than the original weight matrix. The convolution context is split over the two factors. For example, instead of using context (t-2, t, t+2) in the first low-rank factor and no context in the second factor, a context (t-2,t) in the first factor and (t, t+2) in the second factor could be used. During training, a special algorithm that constrains the first of the two factors to be semi-orthogonal is applied. In addition, the architecture includes skip connections similarly to the ResNet architectures. However, in the F-TDNN architecture, the information from the skip connections are concatenated to the output of the previous layer rather than added as in ResNets. We used a publicly available PyTorch implementation[6] that approximately replicates the Kaldi implementation used in Villalba et al. (2020) and trained the model with the WeSpeaker toolkit.

[3] https://huggingface.co/Wespeaker/wespeaker-voxceleb-ecapa-tdnn512.
[4] https://huggingface.co/Jenthe/ECAPA2.
[5] https://github.com/IDRnD/redimnet/blob/master/EVALUATION.md.
[6] https://github.com/cvqluu/Factorized-TDNN/tree/master.

3. Classification back-ends

In this work, we consider cosine scoring and five additional back-ends. The first is Probabilistic Linear Discriminant Analysis (PLDA) (Ioffe, 2006; Prince and Elder, 2007; Kenny, 2010). Gaussian PLDA paired with embedding length-normalization has been the standard classification back-end in the i-vector framework (Dehak et al., 2011), due to good performance, simple training procedure, and fast verification step, but has shown to be less competitive with modern DNN embeddings (Wang et al., 2022).

The second back-end we consider is the Heavy-Tailed PLDA (HT-PLDA) variation of Silnova et al. (2018) and Brümmer et al. (2018). HT-PLDA seeks to address PLDA's limitations when dealing with non-normalized embeddings. Several flavors have been proposed in the literature (Kenny, 2010; Cumani and Laface, 2014a; Silnova et al., 2018; Brümmer et al., 2018). We choose (Silnova et al., 2018; Brümmer et al., 2018) due to its fast training algorithm, paired with a relatively fast verification step.

The third classifier we consider is the Toroidal Probabilistic Spherical Discriminant Analysis (TPSDA) (Silnova et al., 2023), which extends the Probabilistic Spherical Discriminant Analysis (PSDA) approach of Brümmer et al. (2022). These classifiers are a natural choice, since both PSDA and TPSDA focus on modeling embedding directions. PSDA can be interpreted as a probabilistic version of cosine similarity, and TPSDA further combines the PSDA model with PLDA-like speaker and channel subspaces.

Alongside generative classifiers, we also consider discriminative models that have proven effective in the past. These approaches aim at estimating robust scoring functions by training pairwise scoring functions for direct comparison of embedding pairs. We consider the Pairwise Support Vector Machine (PSVM) model (Cumani and Laface, 2014b; Cumani et al., 2013; Cumani and Laface, 2014c; Cumani et al., 2012), which has been shown to provide good results both for older i-vector based systems and for newer DNN-based embeddings (Cumani and Laface, 2014b, 2018; Cumani and Sarni, 2023a; Cumani et al., 2025), and the Discriminative PLDA model of Burget et al. (2011). These approaches typically have slightly larger, although negligible compared to DNN, training computational requirements, but they also allow for fast utterance comparison.

In the following, we briefly recall the individual back-ends.

3.1. Cosine scoring

Cosine scoring computes a speaker verification score between enrollment and test embeddings $\boldsymbol{\phi}_e$ and $\boldsymbol{\phi}_t$ as

$$s(\boldsymbol{\phi}_e, \boldsymbol{\phi}_t) = \frac{\boldsymbol{\phi}_e^T \boldsymbol{\phi}_t}{\|\boldsymbol{\phi}_e\| \, \|\boldsymbol{\phi}_t\|} \ . \tag{1}$$

Although the back-end does not require additional training data, when such data is available it is typically employed to center the embeddings. In this case, cosine scoring is also often paired with Linear Discriminant Analysis (LDA) to remove non-speaker-related embedding components.

3.2. Probabilistic linear discriminant analysis

PLDA models speaker embeddings in terms of Gaussian distributed speaker and channel latent variables. The simplified PLDA model (Kenny, 2010) that we employ represents an embedding as the sum of three terms

$$\boldsymbol{\phi}_{s,i} = \mathbf{m} + \mathbf{U}\mathbf{y}_s + \boldsymbol{\varepsilon}_{s,i} \ , \tag{2}$$

where $\boldsymbol{\phi}_{s,i}$ is the D-dimensional ith embedding of speaker s whereas $\mathbf{y}_s$ is a d-dimensional speaker factor. The vector $\mathbf{m}$ represents the embeddings mean, while $\boldsymbol{\varepsilon}_{s,i}$ represents Gaussian-distributed residual noise. The $D \times d$, with $d \leq D$, factor loading matrix $\mathbf{U}$ can be employed to control the dimension d of the speaker subspace. The model assumes Gaussian priors for the speaker factors and residual noise:

$$\mathbf{y}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \ , \tag{3}$$

$$\boldsymbol{\phi}_{s,i} | \mathbf{y}_s \sim \mathcal{N}\left(\mathbf{m} + \mathbf{U}\mathbf{y}_s, \boldsymbol{\Lambda}^{-1}\right) \ . \tag{4}$$

where $\boldsymbol{\Lambda}^{-1}$ is the noise covariance matrix. The model parameters $\mathbf{m}$, $\mathbf{U}$ and $\boldsymbol{\Lambda}$ can be estimated by Maximum Likelihood (ML) over a training set using an Expectation-Maximization algorithm. For a pair of embeddings $\boldsymbol{\phi}_e$ and $\boldsymbol{\phi}_t$ the scoring function is given by the log-likelihood ratio (LLR) between the same speaker and different speaker hypotheses

$$s(\boldsymbol{\phi}_e, \boldsymbol{\phi}_t) = \log \frac{\int P(\boldsymbol{\phi}_e, \boldsymbol{\phi}_t | \mathbf{y}) P(\mathbf{y}) d\mathbf{y}}{\left(\int (\boldsymbol{\phi}_e | \mathbf{y}) P(\mathbf{y}) d\mathbf{y}\right) \left(\int P(\boldsymbol{\phi}_t | \mathbf{y}) P(\mathbf{y}) d\mathbf{y}\right)} \ , \tag{5}$$

which corresponds to a quadratic form in the embedding pair $(\boldsymbol{\phi}_e, \boldsymbol{\phi}_t)$ (Cumani et al., 2013).

3.3. Heavy-Tailed PLDA

Heavy-Tailed PLDA models introduce additional Gamma-distributed precision scaling factors that make the model covariance matrices utterance-dependent. The original HT-PLDA model (Kenny, 2010) employs hierarchical Gamma priors for both the speaker factors and residual noise precision matrices. While powerful, the model suffers from significant increase in computational requirements. In this work, we consider a simplified HT-PLDA model (Silnova et al., 2018; Brümmer et al., 2018), which pairs a normal-distributed speaker factor with heavy-tailed noise, and employs a Gaussian approximation of the embedding likelihood to

achieve fast and efficient scoring. The model assumes that an embedding can be represented as

$$\boldsymbol{\phi}_{s,i}|\mathbf{y}_s, \lambda_{s,i} \sim \mathcal{N}\left(\mathbf{U}\mathbf{y}_s + \mathbf{m}, \left(\lambda_{s,i}\boldsymbol{\Lambda}\right)^{-1}\right) , \tag{6}$$

$$\mathbf{y}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) , \tag{7}$$

$$\lambda_{s,i} \sim \Gamma\left(\frac{\nu}{2}, \frac{\nu}{2}\right) , \tag{8}$$

where $\lambda_{s,i}$ is an utterance-dependent, Gamma-distributed factor. The model parameters $\mathbf{U}, \mathbf{m}$ and ν can be estimated through Variational Bayes. To simplify scoring, the authors of Silnova et al. (2018) show that the likelihood for $\boldsymbol{\phi}_{s,i}|\mathbf{y}_s$ can be expressed as a $\mathcal{T}$ distribution. As long as the speaker subspace is small compared to the embedding space, this $\mathcal{T}$ distribution can be effectively approximated by a Gaussian, leading to fast scoring. For additional details, we refer the reader to Silnova et al. (2018).

3.4. Toroidal probabilistic spherical discriminant analysis

The TPSDA model (Silnova et al., 2023) shares some similarities with the PLDA model. TPSDA represents a speaker embedding in terms of channel and speaker random variables, as

$$\boldsymbol{\phi}_{s,i}|\boldsymbol{\mu}_{s,i} \sim \mathcal{V}(\boldsymbol{\phi}_{s,i}|\boldsymbol{\mu}_{s,i}, k) , \tag{9}$$

$$\boldsymbol{\mu}_{s,i} = \sum_{j=1}^{m} w_j \mathbf{U}_j \mathbf{y}_{s,j} + \sum_{j=m+1}^{n} w_j \mathbf{U}_j \mathbf{x}_{(s,i),j} , \tag{10}$$

$$\mathbf{y}_{s,j} \sim \mathcal{V}\left(\mathbf{v}_j, \gamma_j\right) \ , j = 1 \dots m , \tag{11}$$

$$\mathbf{x}_{(s,i),j} \sim \mathcal{V}\left(\mathbf{v}_j, \gamma_j\right) \ , j = m+1 \dots n . \tag{12}$$

Here $\mathcal{V}(\boldsymbol{\mu}, k)$ denotes the Von-Mises Fisher (VMF) distribution with mean direction $\boldsymbol{\mu}$ and concentration parameter k. The terms $\mathbf{y}_{s,1} \dots \mathbf{y}_{s,m}$ represent m speaker-dependent, VMF-distributed speaker factors, whereas the terms $\mathbf{x}_{(s,i),m+1} \dots \mathbf{x}_{(s,i),n}$ are $n-m$ utterance-dependent channel factors. The model parameters are the factor loading matrices $\mathbf{U}_j$, the combination weights $w_1 \dots w_m$, the concentration parameter k and the prior parameters $\mathbf{v}_j, \gamma_j, j = 1 \dots n$. To ensure that $\boldsymbol{\mu}$ has unit norm, the TPSDA further constrains the model parameters so that $\mathbf{U}_i^T\mathbf{U}_i = \mathbf{I}$ and $\mathbf{U}_i^T\mathbf{U}_{j\neq i} = \mathbf{0}$, and $\sum_{j=1}^{n} w_j^2 = 1$. In contrast with PLDA, TPSDA assumes (and thus requires) input embeddings to be length-normalized. As for PLDA, the EM algorithm can be employed to estimate the model parameters. The scoring function can also be expressed as the LLR between the same and different speaker hypotheses (Silnova et al., 2023).

3.5. Pairwise SVM

Pairwise SVM (Cumani and Laface, 2014b; Cumani et al., 2013; Cumani and Laface, 2014c; Cumani et al., 2012) directly models the scoring function as a quadratic form of utterance pairs (trials)

$$s(\boldsymbol{\phi}_e, \boldsymbol{\phi}_t) = \boldsymbol{\phi}_e^T\mathbf{A}\boldsymbol{\phi}_t + \boldsymbol{\phi}_t^T\mathbf{A}\boldsymbol{\phi}_e + \boldsymbol{\phi}_e^T\mathbf{B}\boldsymbol{\phi}_e + \boldsymbol{\phi}_t^T\mathbf{B}\boldsymbol{\phi}_t + \boldsymbol{\phi}_e^T\mathbf{c} + \boldsymbol{\phi}_t^T\mathbf{c} + k , \tag{13}$$

where $\boldsymbol{\phi}_e$ and $\boldsymbol{\phi}_t$ are the trial embeddings, $\mathbf{A}$ and $\mathbf{B}$ are symmetric $D \times D$ matrices, $\mathbf{c}$ is a D-dimensional vector and k a scalar. The rationale behind (13) is the formal similarity with the PLDA log-likelihood ratio, which can be represented as in (13) for suitable values of $\mathbf{A}, \mathbf{B}, \mathbf{c}$ and k that depend on the PLDA parameters $\mathbf{U}$, $\boldsymbol{\Lambda}$ and $\mathbf{m}$. The PSVM model, however, does not impose constraints on $\mathbf{A}, \mathbf{B}, \mathbf{c}$ and k, with the symmetry of $\mathbf{A}$ and $\mathbf{B}$ implicitly guaranteed by the PSVM training objective.[7]

Given a dataset with n training embeddings, the PSVM parameters are estimated by minimization of the hinge loss for a pair-wise binary verification task (Cumani et al., 2013, 2012):

$$\frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} \max\left(0, 1 - y_{ij}s(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j; \mathbf{A}, \mathbf{B}, \mathbf{c}, \mathbf{k})\right) , \tag{14}$$

where λ is a regularization hyper-parameter, $\mathbf{w} = \left[vec(\mathbf{A})^T, vec(\mathbf{B})^T, \mathbf{c}^T, k\right]^T$ and the loss term is computed over all possible $n \times n$ embedding pairs, including self-pairs, i.e. $i = j$. The class label y_{ij} equals +1 for target (same speaker), and −1 for non-target (different speaker) pairs. The original PSVM formulation weights the same target and non-target trials, as the objective

[7] It is worth noting that symmetry of $\mathbf{A}$ and $\mathbf{B}$ is not of particular interest, as long as the scoring function remains symmetric in the embeddings. This is guaranteed by the formulation (13) for any matrix $\mathbf{A}$ and $\mathbf{B}$. The original PSVM formulation (Cumani and Laface, 2014b) employs symmetric matrices $\mathbf{A}$ and $\mathbf{B}$, without loss of generality, because the symmetry of the solution is a consequence of the PSVM formulation. Indeed, it can be proven that an optimal solution of the PSVM objective (14) will consist of symmetric $\mathbf{A}$ and $\mathbf{B}$ matrices. To show this, we rewrite the scoring function (13) as $s(\boldsymbol{\phi}_e, \boldsymbol{\phi}_t) = \langle\mathbf{A}, \mathbf{X}\rangle + \langle\mathbf{B}, \mathbf{Y}\rangle + \xi$, where ξ collects terms not depending on $\mathbf{A}$ and $\mathbf{B}$, and $\mathbf{X}$ and $\mathbf{Y}$ are symmetric matrices, with $\mathbf{X} = \boldsymbol{\phi}_e\boldsymbol{\phi}_t^T + \boldsymbol{\phi}_t\boldsymbol{\phi}_e^T$ and $\mathbf{Y} = \boldsymbol{\phi}_e\boldsymbol{\phi}_e^T + \boldsymbol{\phi}_t\boldsymbol{\phi}_t^T$ and $\langle\cdot,\cdot\rangle$ denotes a scalar product. If we consider matrix $\mathbf{A}$, it can be decomposed in its symmetric and antisymmetric components, respectively $\mathbf{A}_S = \frac{1}{2}(\mathbf{A} + \mathbf{A}^T)$ and $\mathbf{A}_D = \frac{1}{2}(\mathbf{A} - \mathbf{A}^T)$, with $\mathbf{A} = \mathbf{A}_S + \mathbf{A}_D$. Since $\mathbf{X}$ is symmetric, we have that $\langle\mathbf{A}, \mathbf{X}\rangle = \langle\mathbf{A}_S, \mathbf{X}\rangle + \langle\mathbf{A}_D, \mathbf{X}\rangle = \langle\mathbf{A}_S, \mathbf{X}\rangle$, since the scalar product of a symmetric and an anti-symmetric matrix is zero. $\mathbf{A}_D$ does not influence the score, and, consequently, the loss part of the objective function. Furthermore, it is easy to verify that $\|\mathbf{A}_S\|^2 \leq \|\mathbf{A}\|^2$, with equality holding if and only if $\mathbf{A}_D = 0$. Indeed, $\|\mathbf{A}\|^2 = \langle\mathbf{A}_S + \mathbf{A}_D, \mathbf{A}_S + \mathbf{A}_D\rangle = \langle\mathbf{A}_S, \mathbf{A}_S\rangle + \langle\mathbf{A}_D, \mathbf{A}_D\rangle + 2\langle\mathbf{A}_S, \mathbf{A}_D\rangle = \|\mathbf{A}_S\|^2 + \|\mathbf{A}_D\|^2 \geq \|\mathbf{A}_S\|^2$, where the equality holds if and only if $\|\mathbf{A}_D\|^2 = 0$, i.e., if and only if $\mathbf{A}_D = 0$, and we used the fact that $\langle\mathbf{A}_S, \mathbf{A}_D\rangle = 0$. It is then easy to verify that, if we assumed that an optimal solution contains a non-symmetric matrix $\mathbf{A}$, then we could build a new solution consisting of the symmetric component of $\mathbf{A}$ which would have the same loss but lower value for the regularization term, contradicting the optimality of $\mathbf{A}$. Similar considerations hold for matrix $\mathbf{B}$.

function naturally tends towards a solution with balanced number of support vectors (Cumani and Laface, 2014b). We observed experimentally that slightly increasing the weight of target with respect to non-target trials leads, on average, to slightly improved results. In this work we scale by a factor 2 the target trials contribution to the loss, and by a factor 1/2 the non-target trials contribution. A similar approach is employed for DPLDA, described in the next section.

Due to the huge amount of trials that can be formed from a single dataset, training a PSVM model with lots of utterances can be expensive. In this work we employ the algorithm of Cumani and Laface (2014b,c), which converges to the optimal PSVM solution but significantly reduces the computational costs through an iterative selection of potential support vectors. We also investigated a GPU-based implementation that employs the same underlying SVM solver of Cumani and Laface (2014b), but recomputes, at each iteration, the loss on the whole set of training pairs (i.e. without trial filtering). Except for an increase in training time with respect to our CPU implementation of the algorithm in Cumani and Laface (2014b), we did not observe differences in the final results.

In contrast with the generative models presented in previous sections, PSVM does not explicitly model a low-dimensional speaker sub-space. However, the regularization term tends to push the model towards solutions where both matrices $\mathbf{A}$ and $\mathbf{B}$ have small eigenvalues, possibly close to zero for embeddings directions that are not relevant for the classification. In this sense, regularization achieves a similar effect as the explicit subspace terms of generative models. To further verify this effect, we extended the PSVM scoring function to allow for explicit dimensionality reduction, by constraining the model parameters[8] $\mathbf{A}$ and $\mathbf{B}$ to be expressed in terms of rectangular matrices $\mathbf{A} = \mathbf{U}_A\mathbf{V}_A^T$ and $\mathbf{B} = \mathbf{U}_B\mathbf{V}_B^T$. We refer to this model as Subspace PSVM (S-PSVM). As we show in Section 4, the explicit subspace does not bring significant improvement with respect to standard PSVM, confirming that regularization is indeed effective in identifying speaker-information-rich directions in the embedding space.

3.6. Discriminative PLDA

The Discriminative PLDA model (DPLDA) (Burget et al., 2011) shares many similarities with the PSVM model, but replaces the hinge loss with binary cross-entropy. The DPLDA objective consists in minimizing

$$\frac{\lambda}{2}\,\|\mathbf{w}\|^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\log\left(1 + exp\left\{-y_{ij}s(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j; \mathbf{A}, \mathbf{B}, \mathbf{c}, \mathbf{k})\right\}\right)\ , \tag{15}$$

where s, λ and $\mathbf{w}$ have the same meaning as in Section 3.5.

In contrast with PSVM, the cross entropy loss entails a small, yet non-zero contribution from all trials. For this reason, the trial filtering-based algorithm of Cumani and Laface (2014b,c) cannot be directly employed for DPLDA. We therefore employed a GPU-based implementation that allows us to train the model over the complete set of training utterance pairs. The optimization is based on the L-BFGS algorithm (Liu and Nocedal, 1989). Given the slightly worse results of DPLDA with respect to PSVM in all setups, and the very limited improvement that explicit subspace modeling brings to PSVM, we did not explore a similar subspace-constrained variant of DPLDA.

4. Experimental results

4.1. Experimental protocol

To assess the quality of the different back-ends for different scenarios we consider several commonly used benchmarks that cover both telephone and non-telephone speech, as well as multiple languages. We divide our evaluation sets into two groups, referred in the following as NIST and non-NIST groups. The former consists of the NIST Speaker Recognition Evaluation benchmarks NIST SRE 2018 (NIST, 2018), NIST SRE 2019 (NIST, 2019), NIST SRE 2021 (NIST, 2021) and NIST SRE 2024 (NIST, 2024), and include both telephone and non-telephone speech. The latter consists of VoxCeleb 1 (Nagrani et al., 2019), SITW (McLaren et al., 2016) and CNCeleb (Li et al., 2023). All NIST datasets include both an official evaluation and an official development set. The former is employed to compute the back-end performance, while the latter is used to perform model selection and hyper-parameter tuning (e.g. subspace dimension) whenever necessary, as detailed in Section 4.4. For non-NIST sets, only SITW includes both an official evaluation and development set. For VoxCeleb, we employ the VoxCeleb Original (Vox1-O) trial list as development set, whereas the evaluation set consists of the VoxCeleb Hard (Vox1-H) trial list. CNCeleb is instead treated as a completely out-of-domain set. Additional details on the number of segments and trials of each set are given in Table 2.

The main evaluation metrics we consider are the minimum Detection Cost Function (min DCF), as defined by NIST for SRE 2018, 2019 and 2024 evaluations, the minimum Cost of Log-Likelihood Ratio (min C_{llr}) (Brümmer and du Preez, 2006; Brümmer, 2010; Van Leeuwen and Brümmer, 2007), evaluated through the PAV algorithm (Zadrozny and Elkan, 2002) on the evaluation set, and Equal Error Rate (EER). To summarize the performance of each system over multiple evaluation sets we employ averages of per-dataset metrics. Due to the large differences in absolute performance on different evaluation sets, we rely on the geometric rather than arithmetic average. While the latter may be more suited to describe mean costs, the former has the advantage that it better represents relative performance gains. A given relative improvement on a dataset, for example, would result in the same relative improvement of the geometric average regardless of the absolute error rate, whereas with arithmetic means a small relative improvement on a difficult benchmark could potentially mask several larger relative improvements on easier sets.

[8] We do not force $\mathbf{A}$ and $\mathbf{B}$ to be symmetrical, since this is not necessary as long as the scoring function remains symmetrical in the two embeddings.

Table 2
Number of speakers, enrollment models, test segments, target and non-target trials for the considered development and evaluation sets.

	Speakers	Enrollment models	Test segments	Target trials	Non-target trials
Development sets					
SRE18-Dev	25	125	1 566	7 830	100 265
SRE19-Dev	58	290	4 066	20 330	618 320
SRE21-Dev	20	158	2 001	15 458	177 793
SRE24-Dev	20	1 139	2 077	110 738	1 064 760
SITW-Dev	119	822	822	2 597	335 629
Vox1-O	40	4 708	4 706	18 802	18 809
Evaluation sets					
SRE18-Eval	188	940	12 135	60 675	2 000 000
SRE19-Eval	137	684	9 515	47 518	2 000 000
SRE21-Eval	182	1 429	17 037	132 038	5 899 731
SRE24-Eval	287	15 998	29 487	1 565 121	6 930 082
CNCeleb	200	196	17 777	17 755	3 466 537
SITW-Eval	180	1 201	1 201	3 658	718 130
Vox1-H	1 190	137 924	135 415	275 488	275 406

4.2. Training sets

Back-end models are trained with either VoxCeleb2 (VC2 back-end in the following) (Okabe et al., 2018), or a mix of VC2 and NIST CTS Superset (Sadjadi, 2021) data (mixed back-end in the following), which consists of the same VC2-only data and additionally the NIST CTS Superset data. For both datasets, we compute a single embedding from all segments belonging to the same session. Since the VC2 dataset includes significantly more sessions than the CTS Superset, to keep the two datasets balanced we randomly selected only 60% of the VC2 sessions, which results roughly in the same number of sessions (and therefore training samples) as the NIST CTS Superset. The total number of speakers, sessions and segments is given in Table 3. The same table reports also the number of target and non-target trials involved in PSVM and DPLDA training.

The full VC2 and CTS datasets were also employed to fine-tune some of the embedding extractors. Since many literature results focus on VoxCeleb data and employ only VC2 as fine-tuning set, we considered both models fine-tuned with VC2 only (VC2 FT in the following), and models fine-tuned with both VC2 and CTS data (Mix FT).

Depending on whether CTS data has been used for training, the different evaluation sets may be considered in-domain or out-of-domain with respect to the front-end and the back-end. In particular, we consider two setups:

- VC2-Only training: in this case NIST and CNCeleb evaluation sets act as out-of-domain data, since CTS data was not used to train neither the front-end nor the back-end.
- CTS + VC2 (mixed) training: in this case both NIST and non-NIST data (except for CNCeleb) can be considered in domain.

We also consider mixed scenarios, where an evaluation set may be out-of-domain with respect to the embedding extractor, but the back-end classifier is trained with domain-matched data.

4.3. Embedding pre-processing

Complex embedding pre-processing can impact the quality of classification back-ends, with optimal results often relying on ad-hoc solutions (Alam et al., 2020, 2022; Cumani et al., 2025). Since we are interested in assessing the effectiveness of different back-ends for generic tasks, in this work we only consider simple pre-processing strategies. For all back-ends except HT-PLDA we center and length-normalize the embeddings. For HT-PLDA we do not apply length-normalization, since HT-PLDA aims at solving the same problem as the length-normalization stage employed in Gaussian PLDA. For cosine scoring we also include an optional (i.e. applied whenever it leads to better results) Linear Discriminant Analysis (LDA) projection stage, aimed at removing non-speaker-related embedding dimensions. It should be noted that, to preserve angles between embeddings in the LDA subspace, our LDA implementation consists in a simple projection, i.e., we do not whiten the within-class covariance of the transformed embeddings. For other back-ends we do not employ LDA, since they already account for a possibly reduced speaker subspace either explicitly (PLDA, HT-PLDA, S-PSVM), or implicitly (PSVM, DPLDA). For FTDNN models, due to few embedding components having almost zero variance, we also applied Principal Component Analysis (PCA) to reduce the embedding dimension to 180.

4.4. Hyper-parameters tuning

Most back-ends include hyper-parameters that influence the quality of the model. Cosine, PLDA, HT-PLDA, TPSDA and S-PSVM require specifying the dimension of the speaker subspace. TPSDA also allows for multiple speaker subspaces, and further requires selecting the number and dimension of the channel subspaces. HT-PLDA allows specifying the degrees of freedom of the model, while

Table 3
Number of segments, sessions and speakers for the NIST CTS (CTS), VoxCeleb2 (VC2) and CTS+VC2 (mixed) training datasets, and corresponding target and non-target trials (pairs) employed for PSVM and DPLDA training.

	Segments	Sessions	Speakers	Target trials	Non-target trials
CTS Superset (CTS)	605 760	86 173	6 867	$\approx 1.55 \times 10^6$	$\approx 7.42 \times 10^9$
VoxCeleb2 (60%) (VC2)	544 806	84 898	5 994	$\approx 1.67 \times 10^6$	$\approx 7.21 \times 10^9$
CTS + VC2 (mixed)	1 150 566	171 071	12 861	$\approx 3.23 \times 10^6$	$\approx 29.26 \times 10^9$

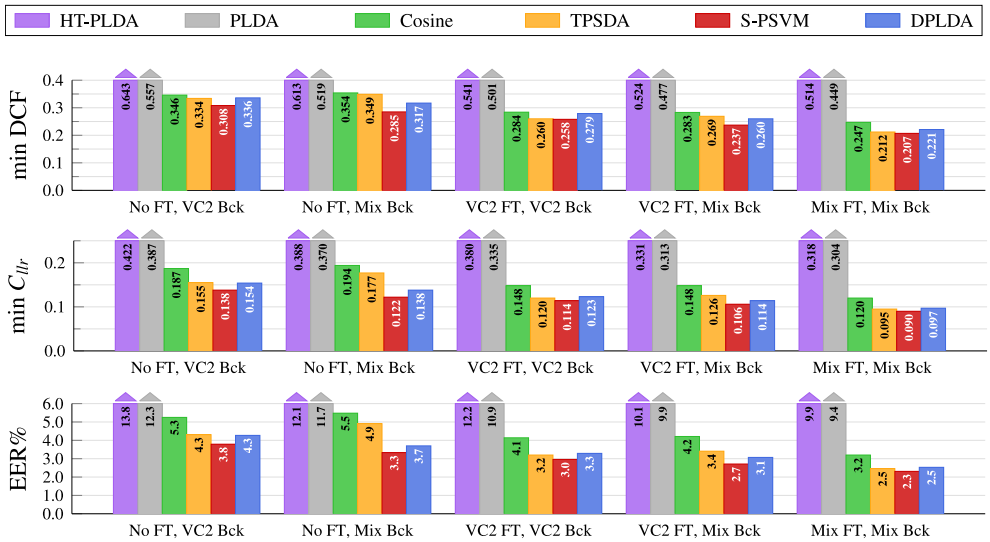


Fig. 1. Minimum DCF, minimum C_{llr} and EER % for different back-ends with SAM-ResNet-100 front-ends, without fine-tuning (No FT), fine-tuned with VC2 (VC2 FT), and with CTS + VC2 (Mix FT) data. As back-ends we consider both models trained with VC2 (VC2 Bck) and CTS + VC2 (Mix Bck) data.

S-PSVM, PSVM and DPLDA require selecting a value for the regularization coefficient. For HT-PLDA, we considered the simple setup with $\nu = 2$ that was employed by the authors in Silnova et al. (2018). Given the poor performance of HT-PLDA compared to Gaussian PLDA with length-normalization, and considering that for large values of ν HT-PLDA would correspond to a Gaussian PLDA with non-normalized embeddings, we did not investigate further values for ν. For TPSDA we employed a similar strategy as in Silnova et al. (2023). We restrict the model to uniform priors, setting all γ_i's to 0. We consider a single speaker subspace, whose dimension is treated as a hyper-parameter. For the channel subspaces, since testing all possible combinations becomes prohibitive, we start with a single channel subspace of size 5 and then we keep doubling the number of subspace factors until we reach a total of 40 channel dimensions over 8 channel terms. The choice was motivated by preliminary analysis that suggested that multiple small channel factors provide better performance with respect to a single large channel term. For all back-ends that include a speaker subspace, we considered dimensions starting from 60, with increments of 30 up to 150, and then increments of 50 up to the embedding size. For (S-)PSVM and DPLDA we employ a regularizer corresponding to the coefficient given in Cumani et al. (2013), multiplied by a scaling factor λ_S that was treated as a hyper-parameter:

$$\lambda = \frac{\lambda_s}{n}\left(1 + \frac{2}{n}\sum_{i=1}^{n}\|\boldsymbol{\phi}_i\|^2\right)^2, \tag{16}$$

where $\boldsymbol{\phi}_i$ is the ith training embedding, and n is the total number of training samples.

Hyper-parameter selection was based on the average results obtained on the SRE18-Dev, SRE19-Dev, SRE21-Dev, SRE24-Dev, Vox1-O and SITW-Dev sets. For each metric, the model that obtained the best average performance was selected. Unless specified, all results on evaluation sets refer to these models.

4.5. Back-end comparison - average results

We start our analysis comparing the average performance of the selected back-ends with different embedding extractors. Our findings are summarized in Figs. 1, 2, 3 and 4, and detailed below.

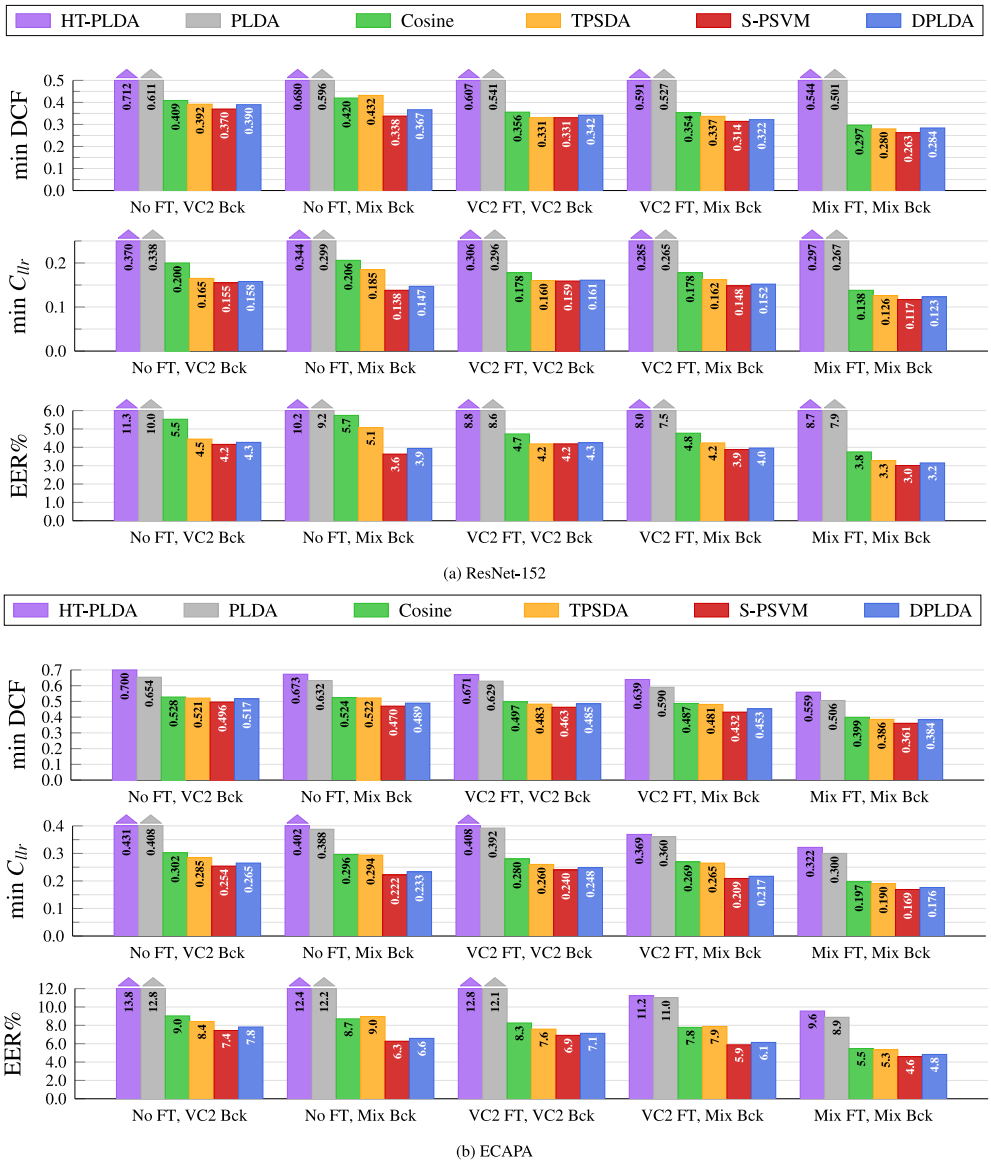


Fig. 2. Minimum DCF, minimum C_{llr} and EER % for different back-ends with different front-ends. For each front-end, we consider both models without fine-tuning (No FT), and models fine-tuned with VC2 (VC2 FT) and CTS + VC2 (Mix FT) data. As back-ends we consider both models trained with VC2 (VC2 Bck) and CTS + VC2 (Mix Bck) data.

4.5.1. SAM-ResNet-100, ResNet-152 and ECAPA

Figs. 1, 2(a) and 2(b) compare the results for the SAM-ResNet-100, ResNet-152 and ECAPA architectures. For all networks, we consider five fine-tuning/back-end training combinations that allow us to analyze the impact of the training data on both fine-tuning and back-end training:

- no DNN fine-tuning, VC2 back-end training
- no DNN fine-tuning, CTS + VC2 (Mix) back-end training
- VC2 DNN fine-tuning, VC2 back-end training
- VC2 DNN fine-tuning, CTS + VC2 (Mix) back-end training
- CTS + VC2 (Mix) DNN fine-tuning, CTS + VC2 (Mix) back-end training

All networks show similar trends. As expected, both PLDA and HT-PLDA provide substantially worse results than cosine scoring. On the other hand, TPSDA, PSVM and DPLDA are typically better than cosine in most, if not all, configurations. DPLDA and TPSDA tend to achieve similar performance on average, while PSVM matches or outperforms them. We can also observe that both cosine

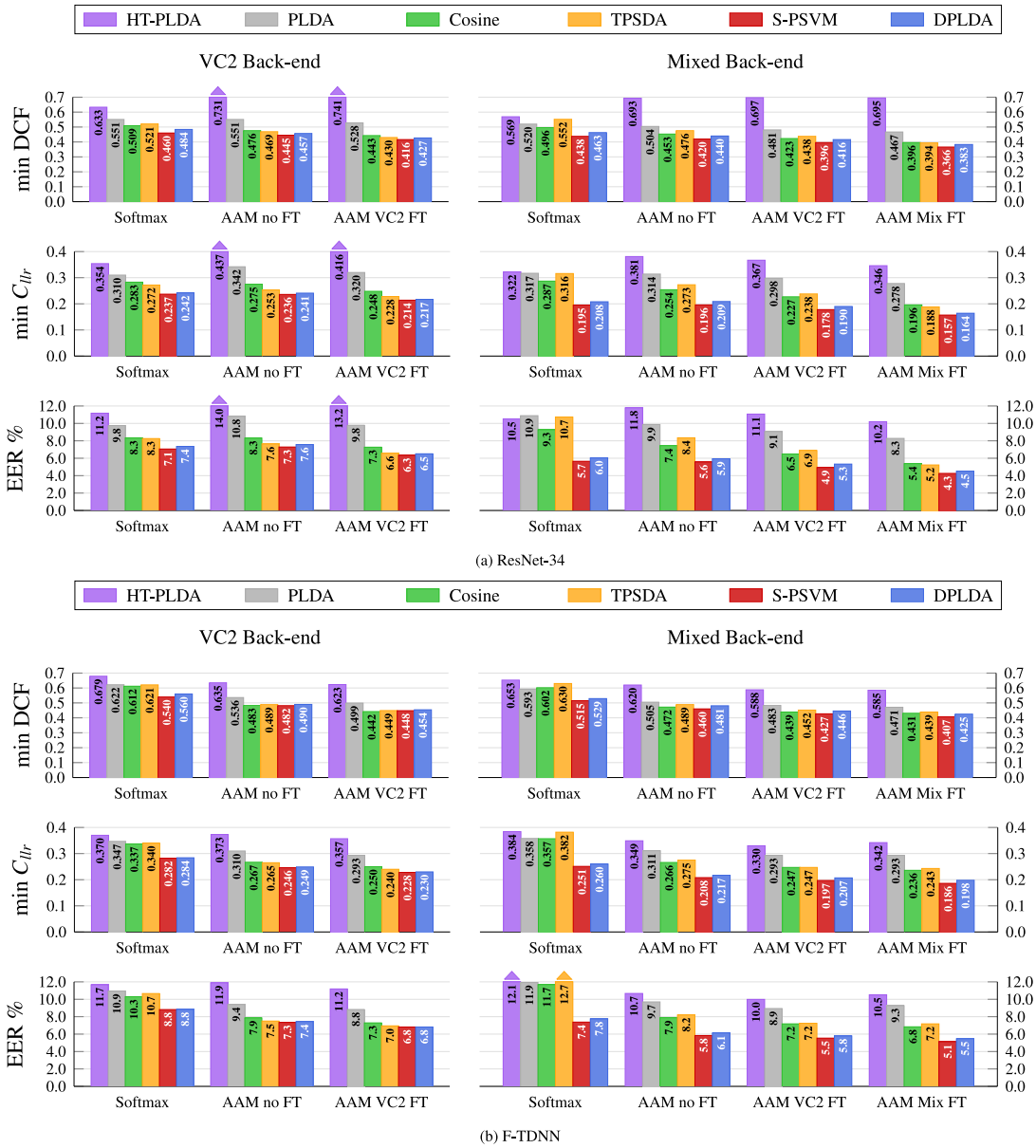


Fig. 3. Minimum DCF, minimum C_{llr} and EER % for different back-ends with different front-ends. For each front-end, we consider models trained with Softmax, and models trained with AAM without (No FT), with VC2 (VC2 FT), and with CTS + VC2 (Mix FT) fine-tuning. As back-ends we consider both models trained with VC2 (VC2 Back-end) and CTS + VC2 (Mixed Back-end) data.

and TPSDA are not significantly affected by the back-end training list, and adding CTS data to the back-end does not improve, and can even slightly degrade, average performance. On the other hand, both discriminative models provide substantially better results when CTS data is added during training. Concerning fine-tuning, we can observe that VC2 FT is beneficial for most models for the SAM-ResNet-100, and it improves cosine and TPSDA performance for ResNet-152 relative to the non-fine-tuned model, but degrades minimum C_{llr} for discriminative models, although PSVM still provides the lowest error rates. Including CTS data further improves the results for all models. Overall, we can conclude that both FT and back-end training can benefit from additional data, although discriminative models seem more effective at leveraging data that was not exposed to the embedding extractor. Among the different methods, PSVM provides overall the best performance for all metrics and all front-ends.

4.5.2. Resnet-34 and FTDNN

Figs. 3(a) and 3(b) show the results of the different back-ends with ResNet-34 and FTDNN architectures. In this case, we consider both networks trained on VoxCeleb2 with Softmax and Additive Angular Margin, the latter both with and without large margin fine

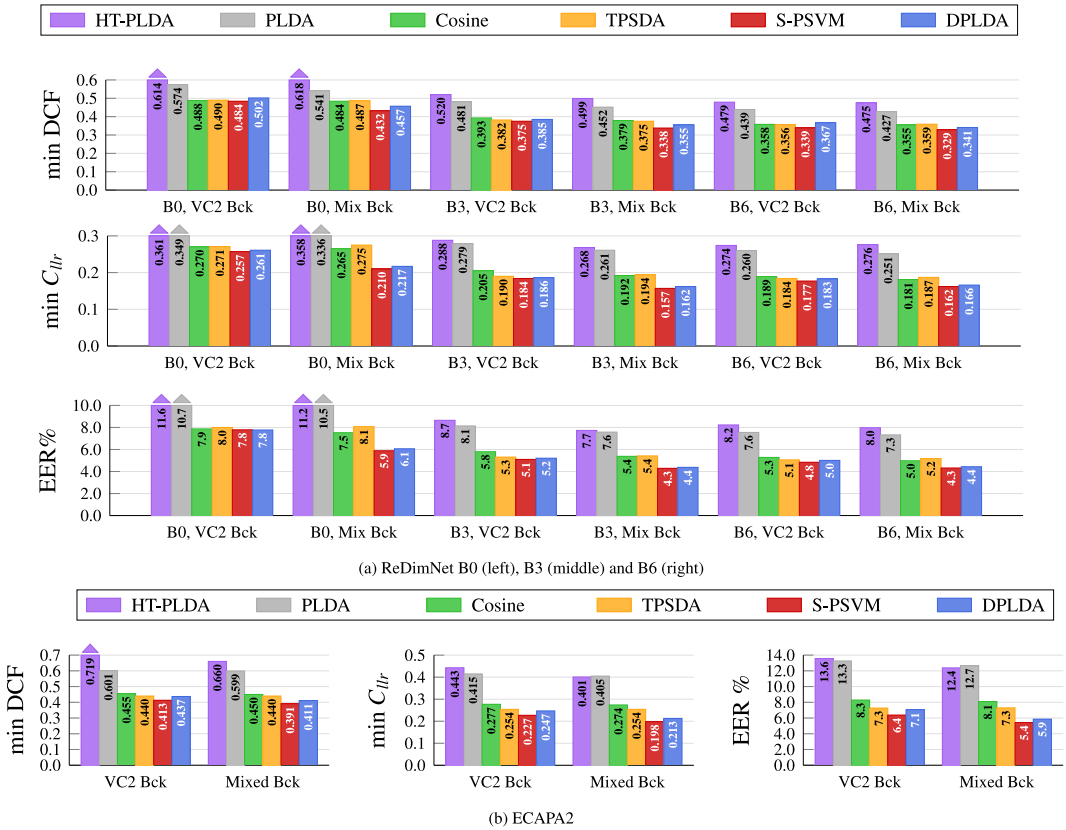


Fig. 4. Minimum DCF, minimum C_{llr} and EER % for different back-ends with off-the-shelf front-ends. For each back-end, we consider both models trained with VC2 (VC2 Bck) and Mixed CTS + VC2 (Mix Bck) data.

tuning. Also in this case, we start observing that both HT-PLDA and PLDA achieve worse results than other methods on AAM-trained networks, with HT-PLDA being the worst model. We recall that HT-PLDA does not include length-normalization, which proves again relevant. For Softmax networks we observe that PLDA is only marginally worse than cosine, and in some cases even better than TPSDA.

With respect to previous front-ends, TPSDA proves substantially less effective, often achieving worse performance than cosine. While cosine benefits in most cases from the additional CTS data, TPSDA provides the best results when trained with just the VC2 set. This suggests that the model performance may depend more than other back-ends on the choice of training data, and the model may not benefit from simply increasing the training set diversity. In Section 4.6 we will analyze this phenomenon more in detail.

PSVM and DPLDA models achieve again best, or close to best, results in all scenarios, with PSVM providing slightly lower error rates than DPLDA. The comparison of Softmax and AAM networks shows that TPSDA models are effective when the networks are trained in a way that emphasizes the role of angles between the embeddings, but perform poorly when this is not the case. We can observe that the PSVM results for the ResNet-34 Softmax and ResNet-34 AAM without fine-tuning are surprisingly close, showing that the AAM objective alone is not, in this case, providing improved performance when embeddings are compared with a robust classifier. This provides a first example that shows that focusing only on cosine distance may lead to a sub-optimal analysis, in some cases, of the relative quality of different embedding extractors.

4.5.3. ReDimNet and ECAPA2

We now consider the remaining off-the-shelf embedding extractors. Fig. 4(a) shows the results for the ReDimNet B0, B3 and B6 models, whereas the ECAPA2 results are given in 4(b), for both VC2-Only and CTS + VC2 back-ends. HT-PLDA and PLDA performance is consistent with our previous findings, with both models providing bad results. Regarding TPSDA, we observe that it struggles to outperform cosine performance for ReDimNet networks, while it improves results for the ECAPA2 architecture. Regarding discriminative models, PSVM gives the best performance for all setups, matching or outperforming both cosine and generative approaches. DPLDA often achieves the second-best results, but suffers in terms of minimum DCF on all the considered setups.

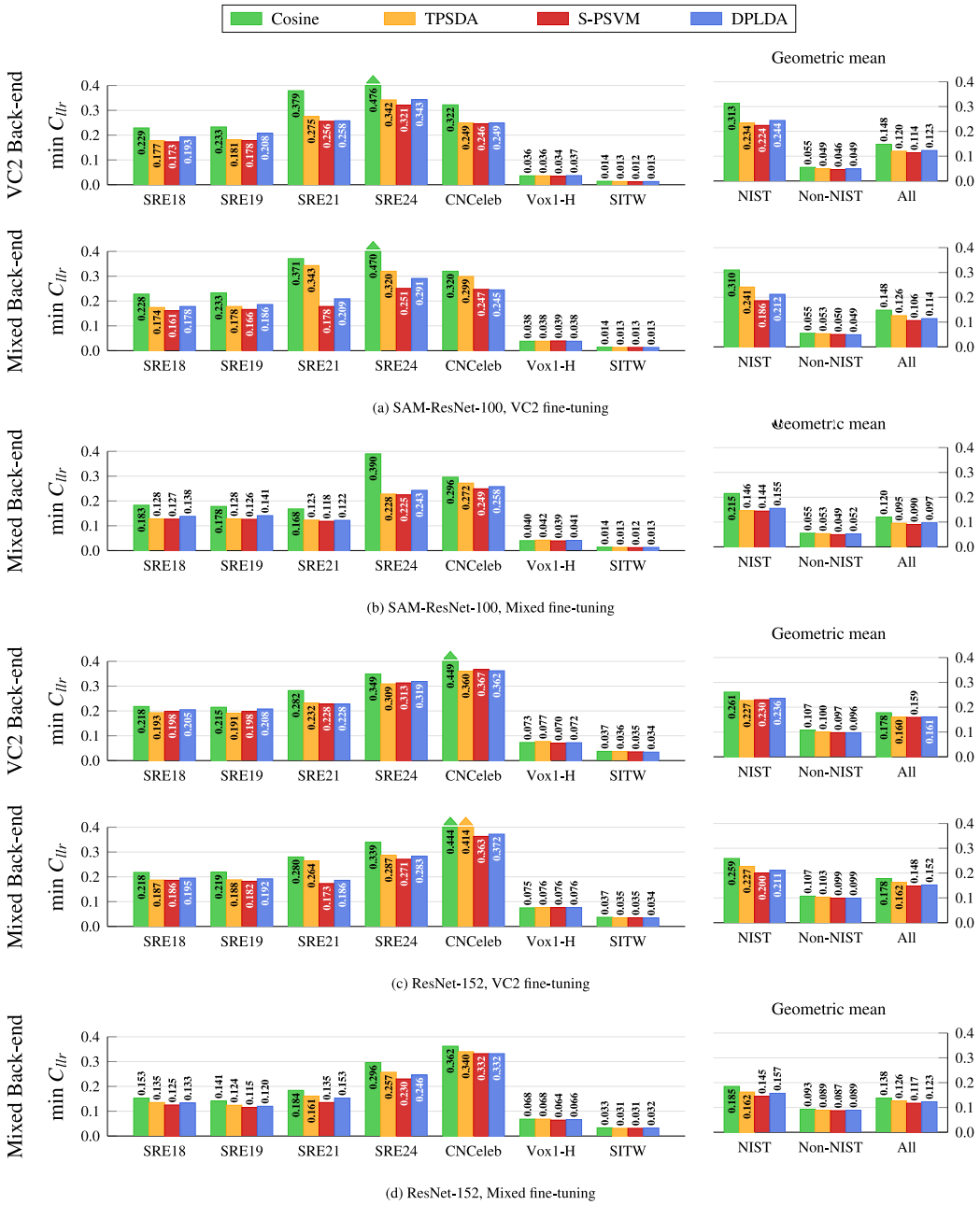


Fig. 5. Individual evaluation set results and corresponding geometric averages, SAM-ResNet-100 and ResNet-152.

4.6. Back-end comparison - individual set results

We have shown that PSVM outperforms cosine distance in almost all the considered scenarios, and TPSDA models can be competitive with PSVM when back-ends are trained with only VC2 data, but, in contrast with PSVM, adding CTS data does not improve, and may even degrade, average performance. In this section we analyze more in details the performance of cosine, TPSDA, PSVM, and DPLDA on individual evaluation sets. We focus mainly on front-ends that include VC2-Only or no fine-tuning at all, considering the two back-end training setups described in Section 4.2, VC2-Only training, and CTS + VC2 (mixed) training, but we also provide individual results for mixed fine-tuning and back-end training for the best performing models, SAM-ResNet-100 and ResNet-152. Figs. 5, 6 and 7 provide results for SAM-ResNet-100, ResNet-152, ReDimNet-B6, ECAPA-2 and ResNet-34 front-ends. We report minimum C_{llr} on the seven individual evaluation sets, and the averages computed on NIST, non-NIST, and all datasets. We can observe a substantial difference in error rates magnitude for NIST and CNCeleb with respect to SITW and Vox1-H, but also

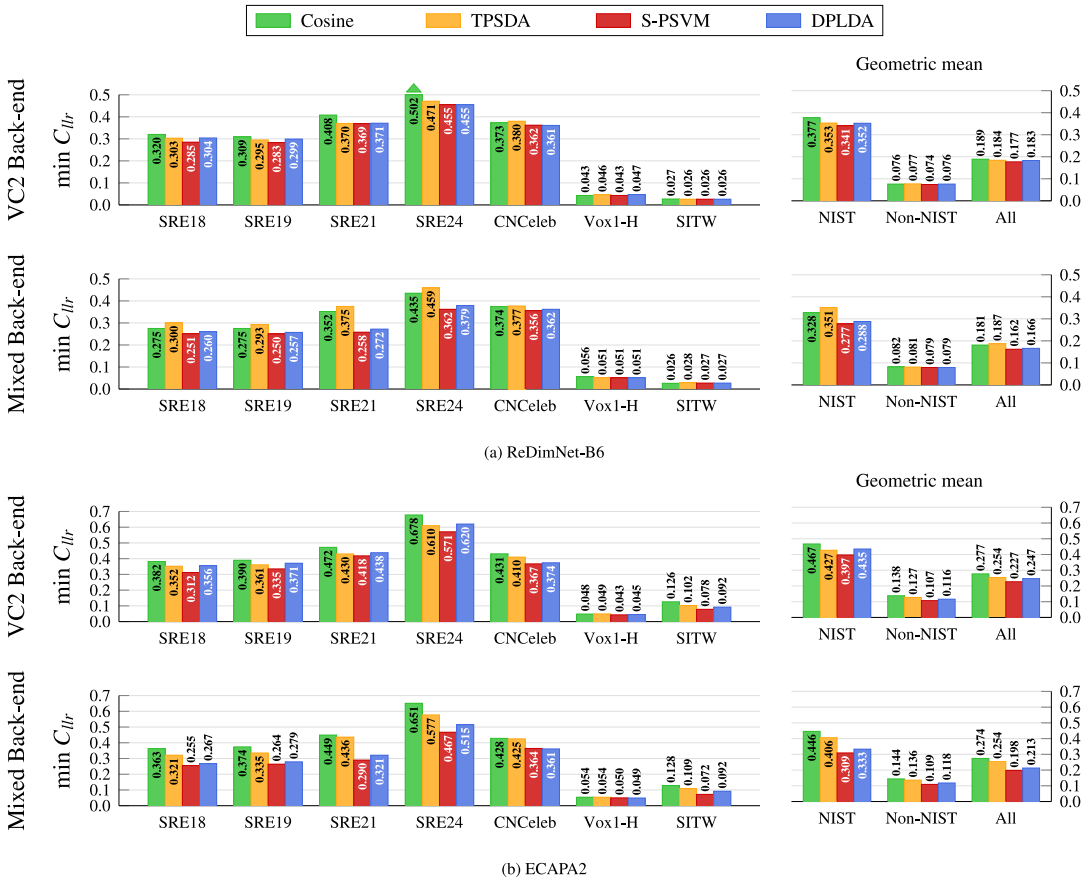


Fig. 6. Individual evaluation set results and corresponding geometric averages, ReDimNet-B6 and ECAPA2 off-the-shelf extractors.

some substantial difference among NIST datasets, which is the reason we chose the geometric rather than the arithmetic mean as an aggregate metric.

Regarding back-ends, results are mostly consistent across the different evaluation sets. When only VC2 data is used for training, cosine scoring provides almost always the worst minimum C_{llr}. TPSDA proves effective for both NIST and non-NIST sets, achieving, on average, similar or slightly worse results than PSVM, but improving substantially over cosine. As concerns discriminative methods, PSVM achieves best or close to best performance across all considered scenarios, whereas DPLDA is usually worse, and comparable to TPSDA. Adding CTS data to the back-end has varying effects on the different classifiers. While most models exhibit a small degradation on non-NIST data, the effects are more pronounced for TPSDA, particularly on CNCeleb. Furthermore, while for PSVM and DPLDA the degradation is more than compensated by a substantial improvement on NIST sets, this is not the case for TPSDA, which can even achieve average worse performance on NIST data, falling short of cosine distance. It is worth noting that the TPSDA performance degradation does not seem related to specific evaluation sets, but rather to the front-ends: for ResNet-34 models, we observe a degradation on most NIST data, whereas for ECAPA2 mixed-trained TPSDA provides overall better results, improving SRE18, SRE19, and SRE24, but degrading SRE21 performance. With ReDimNet-B6 and SAM-ResNet-100 embeddings SRE18 and SRE19 results are not affected by the training list, while mixed training improves SRE24 but degrades SRE21 performance. This phenomenon may seem surprising, as we may expect that adding CTS data to the back-end training list would improve the back-end performance on CTS data. We speculate that the observed degradation may be related to the mismatch between front-end and back-end training data. Since front-ends have not been exposed to CTS data, they may not be able to produce CTS training embeddings that provide a good fit for the TPSDA model assumptions, leading to worse TPSDA models. Indeed, the results with mixed fine-tuning front-ends in Figs. 5(b) and 5(d) show that, once front-end and back-end data are again matched, TPSDA becomes again competitive, achieving PSVM-level performance on SAM-ResNet-100 embeddings.

Fig. 5 allows us to compare the performance of out-of-domain front-end and back-end models, out-of-domain front-ends paired with in-domain (mixed) back-ends, and fully in-domain systems. The VC2 fine-tuned SAM-ResNet-100 and ResNet-152 present, indeed, source mismatch with respect to NIST data, whereas the mixed FT networks can be considered in-domain for the same evaluation sets. Although in-domain (VC2 + CTS) fine-tuning and back-end training proves substantially more effective for all back-ends, we can observe that, for PSVM and DPLDA, it allows for a substantial improvement with respect to out-of-domain (VC2-only)

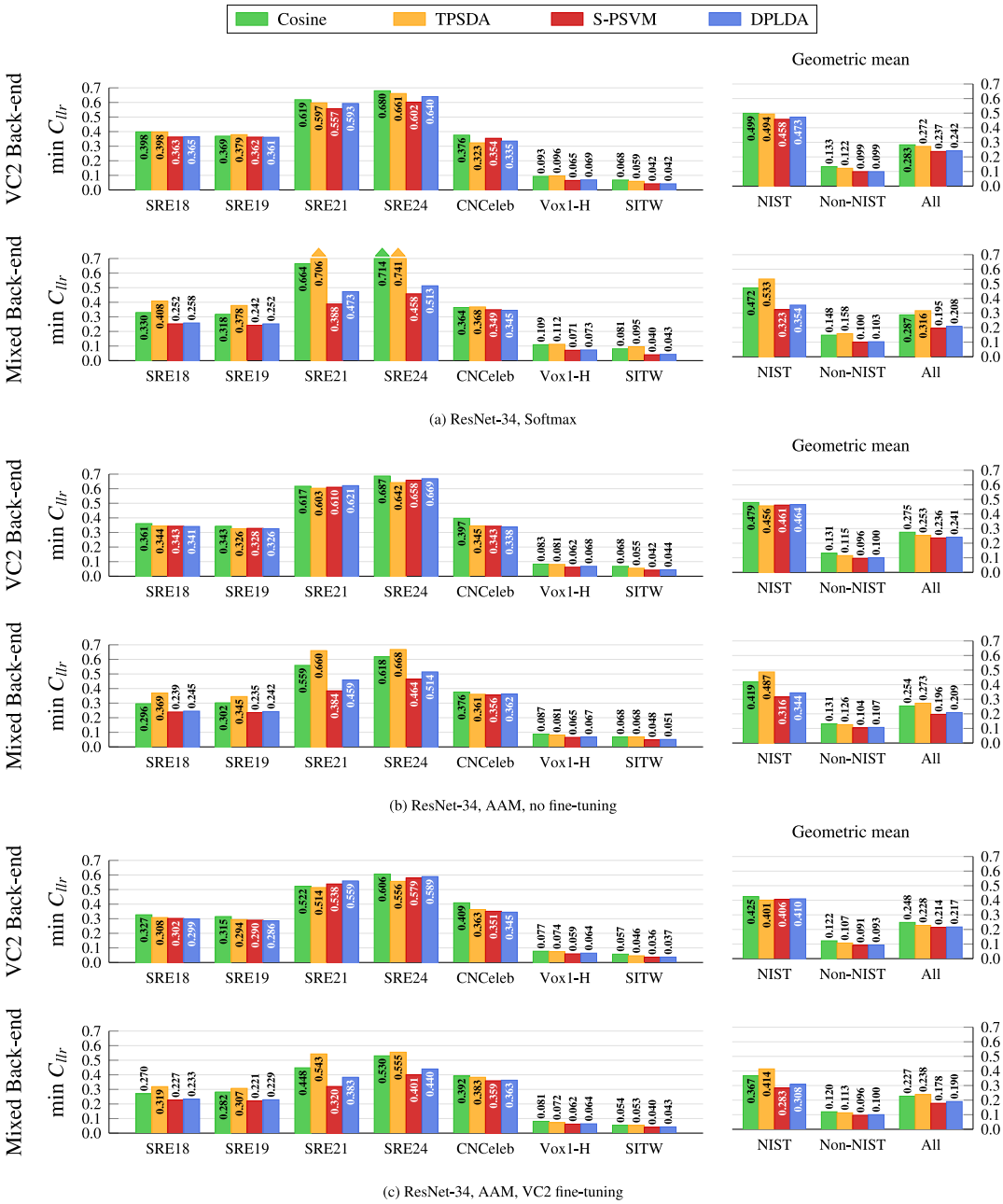


Fig. 7. Individual evaluation set results, and corresponding geometric averages, for ResNet-34 front-ends trained with different losses and fine-tuning strategies.

training. This shows that robust back-ends do not only provide improved results over cosine, but may also leverage additional in-domain data to further improve performance for scenarios where fine-tuning would not be viable due to computational costs.

4.7. Subspace size and hyper-parameters

We now turn our attention to the impact of the speaker subspace size and of the back-end hyper-parameters on the classifiers' performance. Fig. 8 shows minimum C_{llr} as a function of the speaker subspace size. Since PSVM and TPSDA both include additional hyper-parameters, for these back-ends we fixed their values to those of the optimal configuration on the development set (i.e., we vary only the subspace size). Since for TPSDA the sum of the speaker and channel subspace dimensions cannot exceed the original space dimension, the right-most value of each point corresponds to TPSDA models without channel factors, even for those

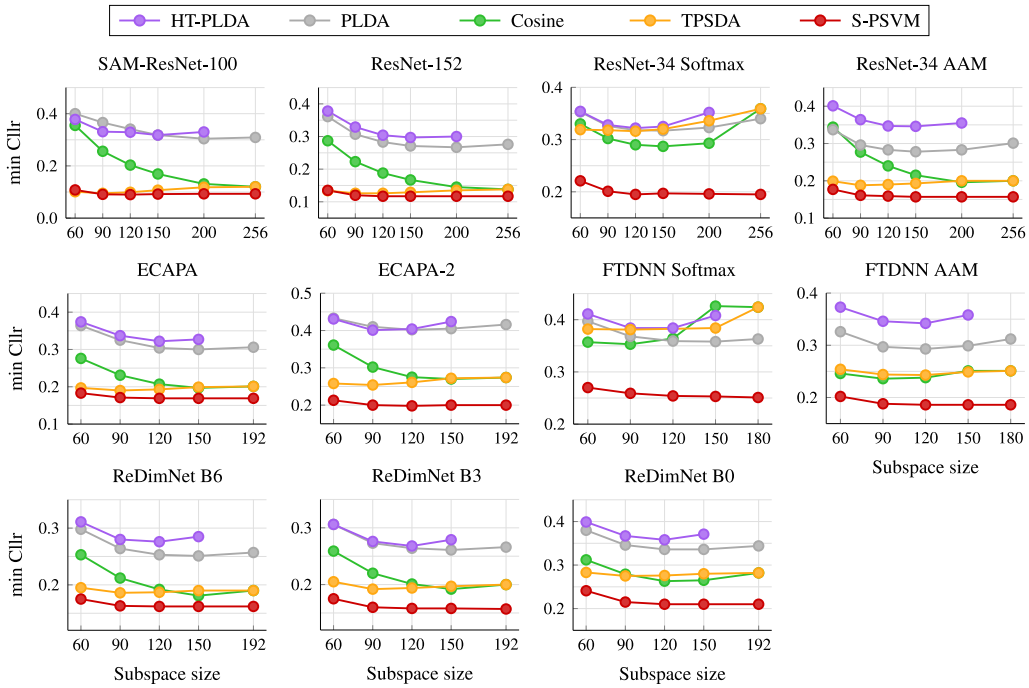


Fig. 8. Results for different front-end/back-end combinations with varying speaker subspace sizes. The SAM-ResNet-100, ResNet-152, ResNet-34 AAM, FTDNN AAM, and ECAPA models employ CTS + VC2 fine-tuning. All back-ends are trained with CTS + VC2 data.

configurations where channel factors provide optimal performance for smaller speaker subspaces. Since the HT-PLDA model of Silnova et al. (2018) employs the speaker subspace complement to estimate the embedding noise, it requires the subspace to be strictly (and possibly significantly) smaller than the embedding space. For this reason the plots do not include HT-PLDA models trained with a full rank speaker subspace.

4.7.1. Cosine and TPSDA

From Fig. 8 we can observe that cosine scoring results are heavily affected by the LDA subspace dimension. For ResNets with AAM loss, ECAPA and ECAPA2 models the LDA reduction is not effective, whereas it gives substantial improvement for Softmax-based models and, to a less degree, also for FTDNN with AAM and ReDimNet front-ends. For TPSDA, on the other hand, small speaker subspaces tend to perform better, as expected (TPSDA with full speaker subspace and uniform priors becomes very similar to cosine scoring (Silnova et al., 2023; Brümmer et al., 2022)). With the exception of Softmax networks, TPSDA achieves, on average, better results than cosine without LDA. However, even for some AAM-based extractors LDA + cosine can, in some cases, obtain slightly better results than TPSDA. Indeed, the two subspace selection methods work in a slightly different way: similarly to TPSDA, LDA performs dimensionality reduction on the length normalized input embeddings, however LDA and TPSDA criteria differ, and, additionally, a second length normalization step is applied after LDA by the cosine scoring procedure, whereas the classification rule of TPSDA depends on the projected embeddings directly (Silnova et al., 2023). We therefore investigated whether applying an additional LDA step can also improve TPSDA. In particular, for the embedding extractors reported in Fig. 9 we considered additional TPSDA models trained with LDA-projected embeddings. For each model, the LDA dimension was selected based on cosine scoring performance. The TPSDA speaker subspace was further optimized on the development sets. We observe that, in almost all scenarios where TPSDA performs worse than LDA + cosine, an additional LDA step allows TPSDA to achieve similar or better performance. However, the LDA + cosine models are still slightly more effective than TPSDA.

Fig. 10 shows the effects of varying both the channel subspace size and the speaker subspace size for the SAM-ResNet-100, the ResNet-34 Softmax and the ResNet-34 AAM front-ends. In the figure, missing columns are due to either the sum of the speaker and channel subspaces being larger than the original embedding space, or to numerical issues of the implementation made available by the authors of Silnova et al. (2023) when the aforementioned sum is close to the embedding size. We can observe that the best results for the former and the latter networks are achieved with small speaker subspaces, without any channel subspace. For the ResNet-34 Softmax front-end, adding a small channel subspace leads to improved results. This suggests that selecting the optimal combination of speaker and channels subspace may be non-trivial for TPSDA models, although we observed that the strategy adopted in Silnova et al. (2023) of optimizing first the speaker subspace and then the channel subspace is typically effective — indeed, we can observe from the figure that the optimal speaker dimension tends to be stable with respect to the channel subspace dimension also in our experiments.

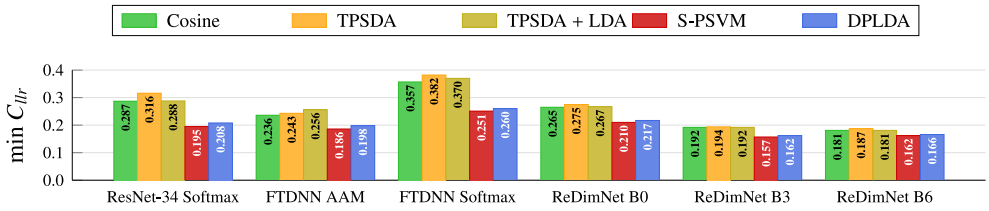


Fig. 9. Comparison of TPSDA and TPSDA with an additional LDA pre-processing step.

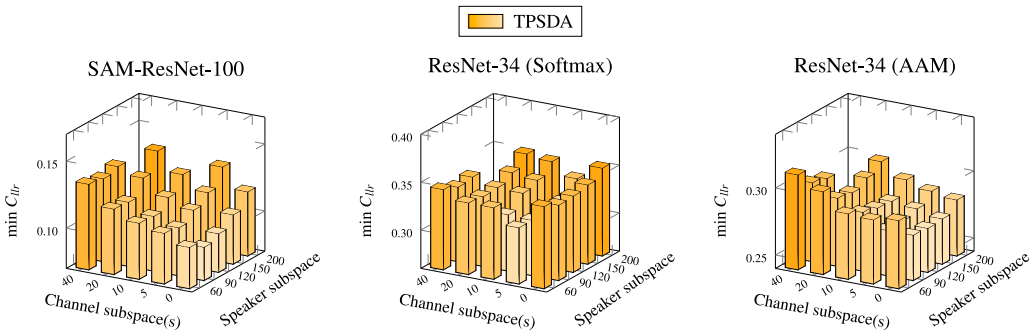


Fig. 10. TPSDA results as a function of the speaker and channel subspace(s) dimension.

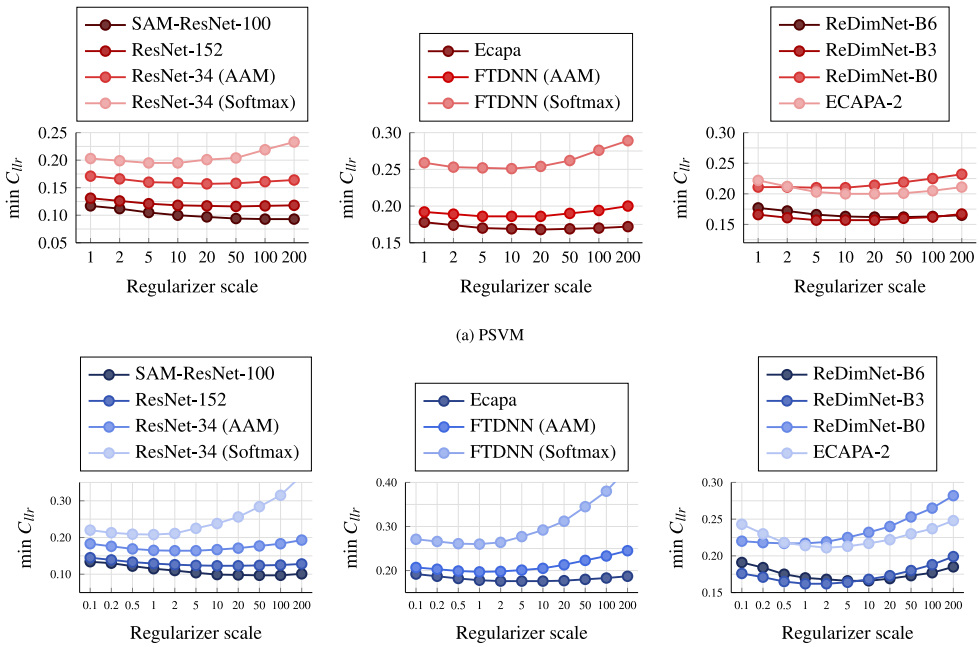


Fig. 11. Average minimum C_{llr} on the evaluation sets as a function of the PSVM and DPLDA regularization scaling coefficient λ_S.

4.7.2. S-PSVM and DPLDA

PSVM, S-PSVM and DPLDA all include a regularization factor that can affect the performance of the resulting model. Fig. 11 shows the impact of different regularizers. On the x-axis we plot the regularization scaling coefficient λ_s in (16). We can observe that while the optimal value differs from front-end to front-end, a simple grid search allows identifying effective values for all the embedding extractors. We also observe that, in contrast with the findings in Cumani et al. (2013), large values of λ_s tend to provide better results for PSVM, while $\lambda_s \approx 1$ seems more effective for DPLDA-based models. Figs. 8 and 12 confirm that explicit subspace modeling is not required by PSVM. Indeed, as long as the speaker subspace dimension is not too small, the S-PSVM models achieve

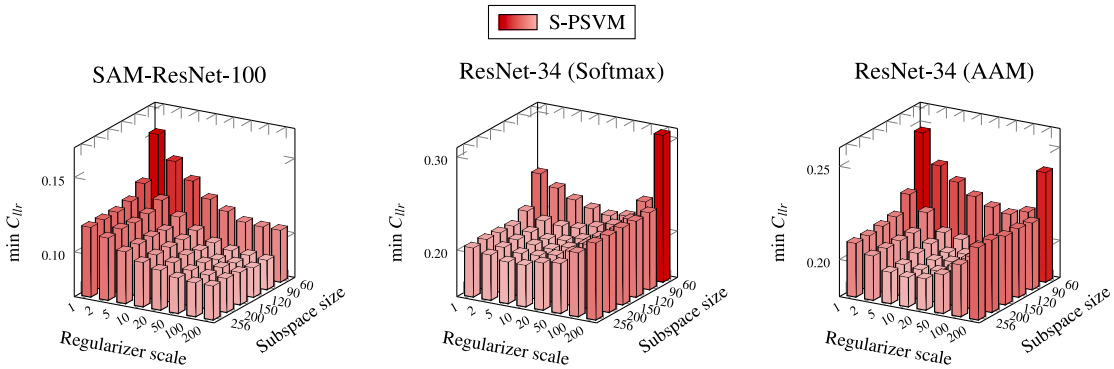


Fig. 12. S-PSVM results as a function of the speaker subspace dimension and the regularization scaling coefficient λ_S.

very close performance as the standard PSVM for all the considered regularizer coefficients. On the contrary, explicitly forcing the subspace to be of small dimension may hinder PSVM performance, as it may discard useful speaker information.

4.7.3. Per-dataset hyper-parameters optimization

In previous sections, we have employed cross-validation to select a configuration for each back-end that would work optimally on our development set on average. In this section, we analyze how per-dataset hyper-parameter tuning may impact performance. We consider the two best-performing front-ends, SAM-ResNet-100 and ResNet-152, in the VC2-Only and CTS + VC2 fine-tuning setup, with back-end training employing CTS + VC2 (mixed) data. Fig. 13 compares the geometric average and individual evaluation set results for these configurations. In particular, for each front-end, we compare the results of the configuration that achieved the best average performance on the development sets with those obtained by selecting the best configuration for each individual evaluation set, corresponding to a lower bound on the back-end performance. We can observe that the average results are similar, suggesting that, while per-dataset hyper-parameter tuning may be effective, the improvements are typically small and often restricted to individual datasets and specific front-ends — for example, we can see a substantial gain for TPSDA for the SAM-ResNet-100 with VC2 FT on SRE-24 and SRE-21, but the same improvement does not occur for other back-ends, other evaluation sets or even to the SAM-ResNet-100 with CTS + VC2 FT. We can also observe that, in general, the relative performance of the different back-ends is preserved, on average, whether a single setup across all datasets or per-dataset-optimized setups are used.

4.8. Score normalization and domain adaptation

Score normalization can significantly impact the performance of speaker verification systems in presence of mismatched and/or multiple test conditions (Kenny, 2010; Matějka et al., 2017; Colibro et al., 2017; Cumani and Sarni, 2022; Alam et al., 2020). In particular, adaptive score normalization (Sturim and Reynolds, 2005; Cumani et al., 2011; Karam et al., 2011) can leverage unsupervised datasets that do not require multiple repetitions per speaker and can improve in some, although not all cases (Cumani and Sarni, 2023b), the recognition performance. In this section, we analyze the impact of one of the most successful approaches, Adaptive S-Norm (Cumani et al., 2011; Karam et al., 2011), as implemented in[9] (Matějka et al., 2017). Alongside AS-norm we also consider a second normalization approach, adaptive data normalization (AD-norm) (Cumani and Sarni, 2023b). AD-norm can be interpreted as an adaptive domain adaptation approach that employs an adaptive cohort to perform a centering of the enrollment and test embeddings. The normalization cohort consists of about 2400 segments from VoxCeleb2, 1200 segments from the CTS Superset, and about 1200 utterances taken from the SRE-18 Unlabeled set (NIST, 2018). For each back-end, we employ the hyper-parameter configuration that provided the best performance without score normalization. For each metric, the cohort size is selected according to the average results on the development set, considering a small set of possible values (200, 400 and 600 utterances). Results are shown in Fig. 14 for the two best front-ends, SAM-ResNet-100 and ResNet-152, and for ReDimNet-B6, ECAPA and ResNet-34 front-ends. For SAM-ResNet-100 and ResNet-152 we consider both the mixed-FT and back-end training scenario, which simulates a matched training condition (except for CNCeleb data), and VC2-Only FT/back-end training, which represents a use-case with partial mismatch between training and evaluation conditions. For ResNet-34 we report both AAM and Softmax results. Due to space constraints, we do not report the results for other front-ends, although the final considerations are similar.

We observe that AS-norm is effective for all methods, in particular for the minimum DCF metric. The largest relative improvement is obtained for PLDA and HT-PLDA, which, however, still under-perform other approaches. Cosine scoring also shows slightly larger improvement than TPSDA, PSVM or DPLDA, reducing a bit the gap between cosine and the better performing TPSDA and PSVM models. However, even after score normalization TPSDA and PSVM are still substantially better than cosine, with the latter

[9] The original AS-norm of Cumani et al. (2011) computes the adaptive cohort in a slightly different way than the AS-norm given in Matějka et al. (2017). While the former provides marginally better results, the latter was chosen due to being significantly faster at scoring time.

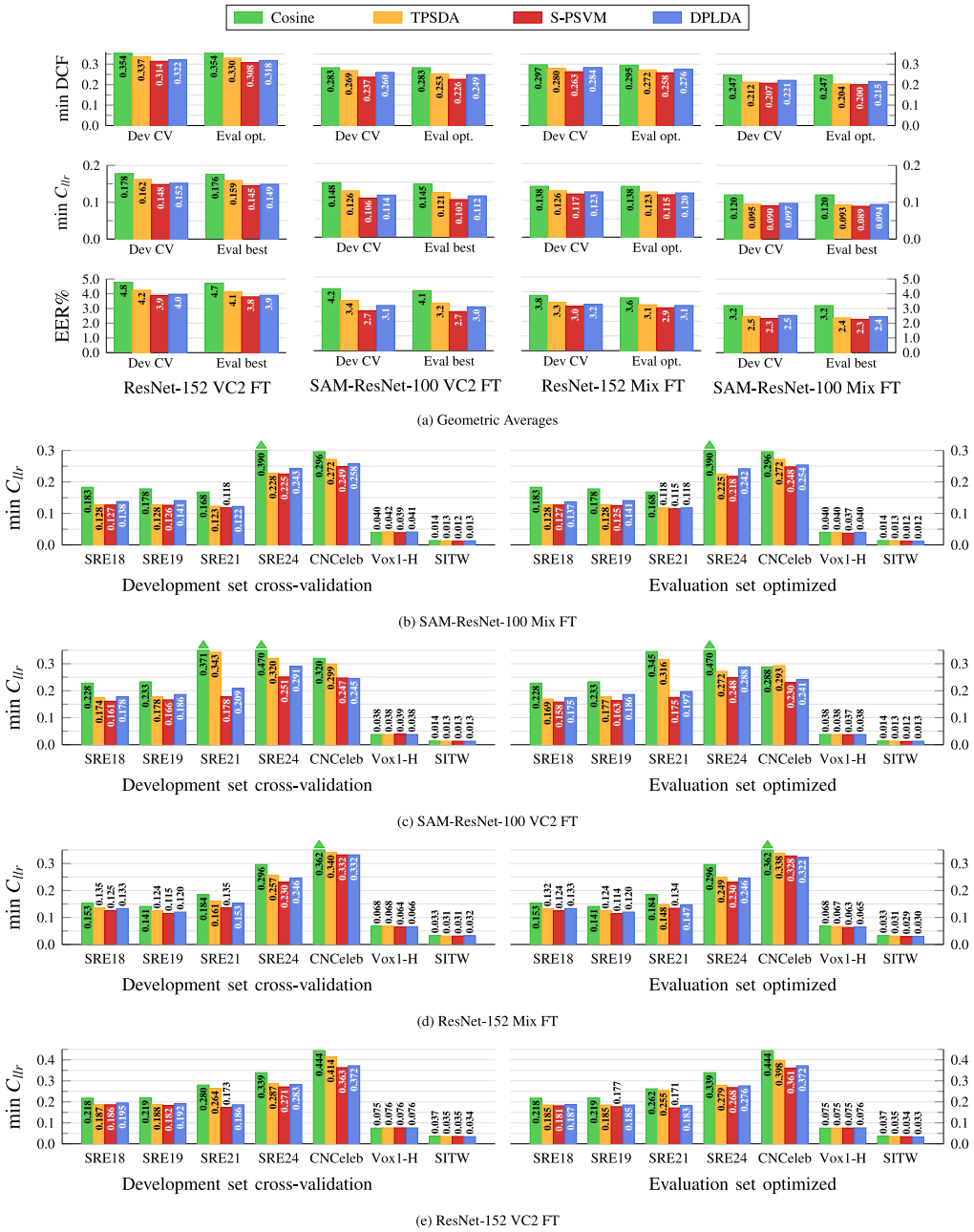


Fig. 13. Effects of per-dataset hyper-parameter tuning. Development set cross-validation (Dev CV): hyper-parameters are selected by cross-validation optimizing the average development sets results. Evaluation set optimized (Eval best): for each evaluation set the hyper-parameters that provide the best results are employed.

outperforming the former by a similar margin. Similar considerations hold for AD-norm. Our results show that the approach is also effective, especially for more robust back-ends. For cosine scoring we observe that AD-norm is on average slightly less effective than AS-norm, whereas for PSVM we typically observe very similar or, in some cases, even slightly better results than with AS-norm.

4.9. Discussion

Our results show that, despite its wide adoption, cosine scoring is often a sub-optimal back-end choice for comparing speaker embeddings, and better back-ends improve performance in many scenarios. Indeed, as shown in Fig. 15(a), if restrict ourself to cosine, TPSDA, PSVM and DPLDA, the former ranked last in more than 60% of the 259 individual evaluation set experiments that

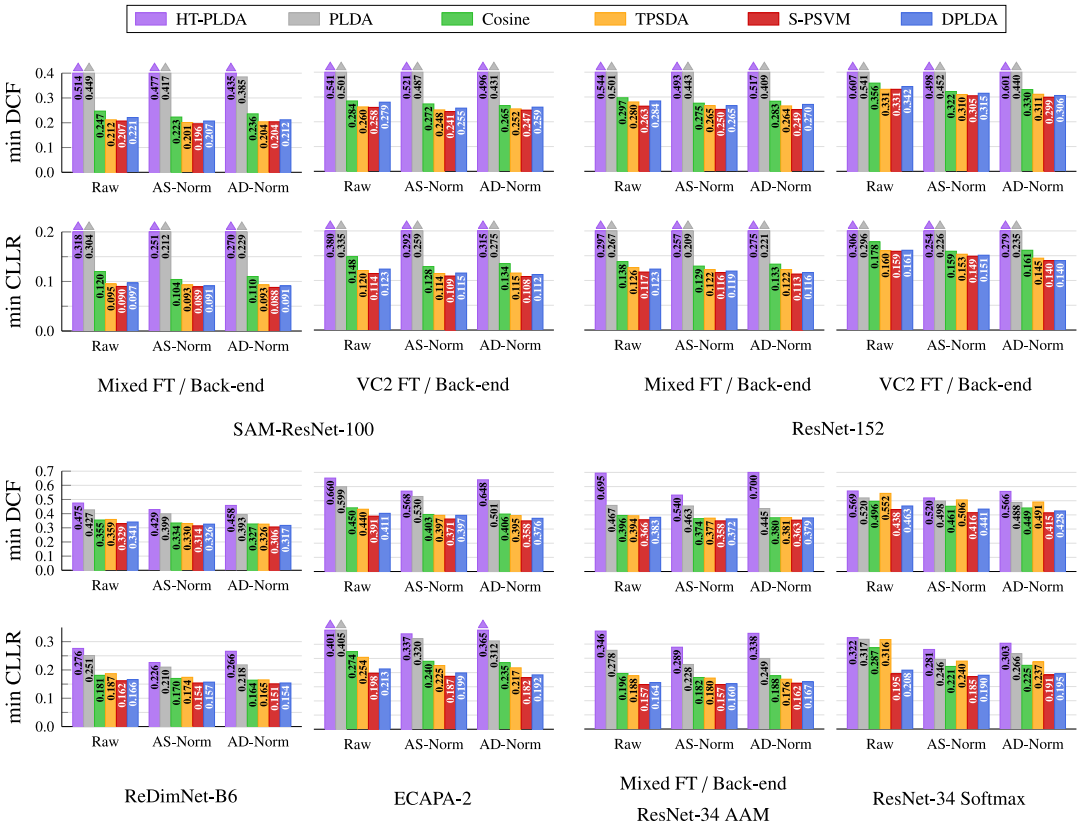
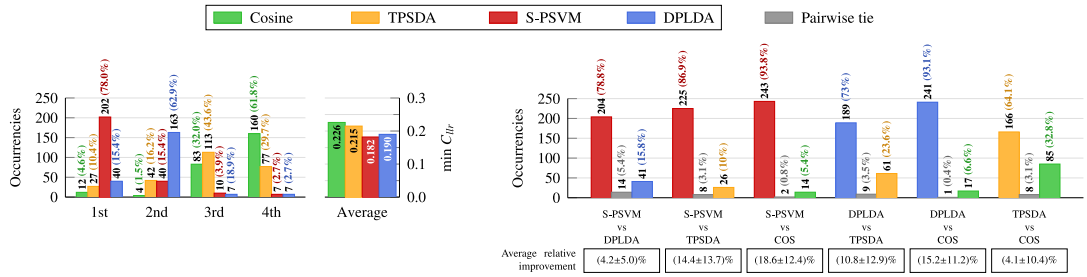
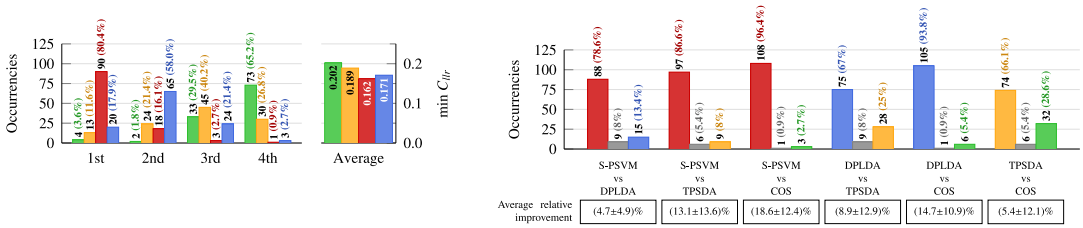


Fig. 14. Minimum DCF and minimum C_{llr} for SAM-ResNet-100, ResNet-152, ReDimNet-B6, ECAPA-2 and ResNet-34 front-ends, without score normalization, with AS-norm and with AD-norm.

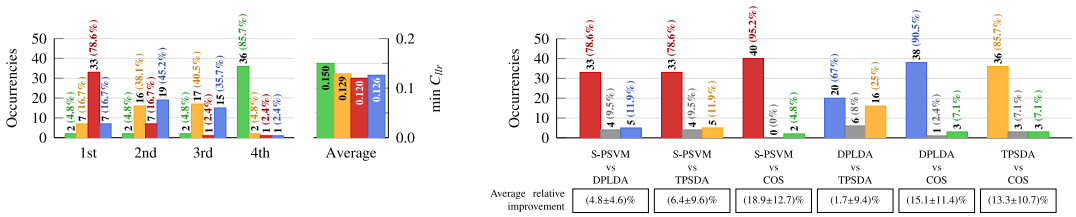
we analyzed, corresponding to the front-end architectures, and front-end and back-end training setups that we summarized in Figs. 1 to 4, showing a relative degradation of average minimum C_{llr} of 24% with respect to the best back-end, S-PSVM. These effects are even more pronounced for the two best performing front-end architectures, SAM-ResNet-100 and ResNet-152, as shown in Fig. 15(c). The worse results of cosine scoring with respect to TPSDA, PSVM and DPLDA can be explained by the strong but not necessarily accurate assumptions of the back-end, which implicitly assumes an isotropic model in which all embedding components are weighted the same. Cosine scoring can indeed be interpreted as a non-probabilistic version of a Spherical PLDA (Sholokhov et al., 2023) model, which assumes isotropic covariance matrices for both the speaker and channel factors and a full-sized speaker factor. While LDA is, in some cases, effective, the LDA assumptions do not properly account for the angular characteristics of DNN embeddings. Indeed, LDA proves effective mainly for Softmax networks, as shown in Fig. 8. TPSDA, on the other hand, models directional data with greater flexibility, allowing us to constrain speaker factors in a way that is consistent with the angular loss criterion of the embedding extractors, which emphasize the role of angular distances, and contains trainable parameters that describe nuisance effects. Concerning PSVM, the improvements over cosine can be explained through the combination of a discriminative training criterion, the additional flexibility of the scoring function, which includes cosine scoring as a special case as long as unit-norm embeddings are employed, and the presence of a regularization coefficient that induces an effect similar to that of TPSDA subspace constraints. The relative performance of PSVM with respect to TPSDA may be explained by the weaker assumptions of PSVM on the underlying data distribution (we note, however, that although the PSVM scoring function may formally represent both PLDA and cosine models, this is not true for TPSDA models, and that in some cases TPSDA proves competitive with discriminative approaches). Finally, DPLDA worse performance with respect to PSVM can be explained by the characteristics of the training framework. Both PSVM and DPLDA are trained over embedding pairs, which are assumed to be independent. However, when pairs are formed by combining all utterances of a training set, the independence assumption is not met. The work (Cumani and Laface, 2014b) shows that the number of relevant pairs for PSVM grows linearly with the number of speakers in the training set, rather than quadratically as the total number of trials. It is reasonable to expect that also for DPLDA there will therefore be a small number of relevant non-target trials, and a large number of easy (i.e. far from the separation boundary), strongly correlated non-target trials. Easy trials introduce a contribution to the logistic loss employed by DPLDA that is small for each trial, but substantial once all the trials are pooled together. Since trials are correlated, it becomes hard to balance the contribution of easy and difficult trials. The PSVM loss, on the other hand, achieves naturally a balance, since easy non-target trials do not affect the corresponding hinge loss regardless of their number, and difficult non-target trials are proportional to the number of speakers, and thus of target trials.



(a) Summary of individual evaluation set results corresponding to all analyzed front-end / back-end combinations, described in Figures 1 to 4, for a total of 259 individual benchmarks.



(b) Summary of individual evaluation set results corresponding to the front-end / back-end combinations in Figures 5 to 7, for a total of 112 individual benchmarks.



(c) Summary of individual evaluation set results corresponding to the ResNet-152 and SAM-ResNet-100 configurations in Figure 5, for a total of 42 individual benchmarks.

Fig. 15. Summaries of relative back-end rankings for cosine, TPSDA, S-PSVM and DPLDA on individual evaluation sets, and corresponding global geometrical average minimum C_{llr}. Left: number and percentage of times each back-end achieves a given rank. In case of ties (same result up to the third decimal digit) we assign the same rank to both back-ends (thus the total number of back-ends achieving a given rank may differ from the total number of considered benchmarks). Right: Pairwise comparisons. For each back-end pair we report how many times each model achieves better results, and the number of ties. The best performing classifier is on the left of each group. Below each group we report the average improvement, measured as the arithmetic mean of all relative improvements, computed on each evaluation set for each front-end/back-end combination, of the best performing back-end over the worst one, and the corresponding standard deviation.

An apparently appealing property of cosine is that it is simple to deploy as it does not, in its simplest form, require neither an additional training step, unless LDA is employed to pre-process the embeddings, nor complex scoring, as, for example, TPSDA. However, we observe that the good performance of cosine scoring relies on effective training of the embedding extractor. Both DNN training and fine-tuning procedures play an important role, and the training material employed for fine-tuning can have a significant impact on cosine performance. In this sense, cosine scoring training is essentially hidden in the embedding extractor training step. Our results show that the analyzed back-ends are able to outperform cosine scoring when trained on the same data that was employed for fine-tuning, and our experiments on non-fine-tuned embedding extractor variants that were trained on VC2 only data (ResNet-34, ECAPA, FTDNN) suggest that similar results hold also when the original DNN training data is employed to train a robust back-end for non-fine-tuned front-ends. The robust back-ends we analyzed do not therefore require more training material with respect to cosine once we consider the full training pipeline. From the point of view of computational complexity, all the presented back-ends can be trained on modern CPUs in just few minutes in the worst case, except for DPLDA models that can nevertheless be trained on a GPU in few tens of minutes. The impact in terms of computational time is therefore negligible with respect to fine-tuning, and, as we have shown in Section 4.6, these back-ends can even be employed as an alternative, although less effective on their own, to fine-tuning for those scenarios where training material is available but re-training the embedding extractor is not viable due to computational costs. Regarding scoring time, TPSDA has slightly larger computational costs than cosine, however these costs can still be considered negligible compared to embedding extraction. PSVM and DPLDA scoring can be implemented with a test cost that is essentially the same as cosine scoring (Cumani and Laface, 2019). Overall, these observations show that cosine scoring deployment benefits may actually be considered marginal: robust back-ends can be trained with no additional training material,

and with negligible computational costs compared to training the embedding extractor, and still provide substantial gains both in matched and mismatched domain scenarios.

The weak performance of cosine raises some concerns on its usage as a standard back-end to compare different architectures. Besides the results mentioned in Section 4.5.2, which show that relying on cosine scoring alone may indeed bias the comparison of embedding extractors trained with different loss functions, we can observe that, if we consider individual benchmarks, cosine can lead to misleading conclusions in other different cases that share little similarities. For example, if we compare the SAM-ResNet-100 and ResNet-152 models with mixed fine-tuning and back-end training lists on the SRE24 benchmark (Figs. 5(b) and 5(d)), we observe that cosine scoring would suggest the former to be about 30% worse than the latter. However, if we compare the same extractors with a PSVM model, which achieves substantially better results on both, the two extractors become much closer, with the former actually becoming the most effective by a small margin. Similar considerations hold comparing the SAM-ResNet-100 with VC2 fine-tuning and mixed back-end with the ResNet-152 models following the same setup. In this case, for all NIST sets except SRE21 the former network seems worse when cosine scoring is employed as back-end, but actually achieves better performance when cosine is replaced by more effective PSVM models. To give another example, we consider the ReDimNet-B6 and ResNet-34 AAM, fine-tuned on VC2. With a mixed back-end the former achieves around 2% better results on SRE19 when cosine scoring is employed. With PSVM, however, the latter proves to be more than 11% better. If we consider VC2-only back-ends, on CNCeleb the ReDimNet-B6 seems almost 5% better with cosine, but with PSVM the ResNet-34 embeddings achieve better, albeit only slightly, results. With the same networks we observe the opposite behavior on SRE18: with cosine scoring ReDimNet-B6 seems 2% worse than ResNet-34, but with PSVM the former becomes more than 5% better. Fig. 15 shows, for each back-end pair excluding PLDA or HT-PLDA, the relative performance of the considered classifiers, and provides the relative average improvement of the best performing of each couple, measured as the arithmetic average of relative improvements of the best-performing (on the left of each pair) back-end with respect to the worst one, computed over the individual benchmarks that we analyzed, and the corresponding standard deviation. These results confirm that cosine scoring is typically worse than other back-ends and relative gains can have substantial variations from use-case to use-case, thus cosine scoring cannot reliably predict the performance of better back-ends. These facts, together with the explicit examples we provided, raise substantial concerns on the systematic usage of cosine scoring to compare front-end architectures.

5. Conclusions

We have critically analyzed the role of back-end classification in the DNN embeddings speaker recognition framework, showing that robust back-end models can substantially improve performance over cosine scoring across a multitude of supervised DNN embedding extractors and different use cases. Among the considered back-ends TPSDA and PSVM prove the most effective, with the latter typically achieving similar or slightly better results than the former. PSVM shows higher robustness to the characteristics of the DNN embeddings and better ability to leverage additional back-end training data. TPSDA, on the other hand, seems effective in scenarios where DNNs are trained with angular losses, and there is a strong match between the DNN and the back-end training material, providing a competitive alternative to discriminative methods. Overall, our experiments highlight the importance of proper back-end modeling also in the DNN embedding framework, and suggest that the current over-reliance on cosine scoring may not only lead to sub-optimal performance, but may even bias the evaluation of potential DNN embedding architectures and training strategies.

CRediT authorship contribution statement

Sandro Cumani: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Sara Barahona:** Writing – review & editing, Writing – original draft, Software, Methodology. **Johan Rohdin:** Writing – review & editing, Writing – original draft, Software, Methodology. **Anna Silnova:** Software. **Oldřich Plchot:** Writing – review & editing, Writing – original draft, Supervision, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Computing on IT4I supercomputer was supported by MoE through the e-INFRA CZ (ID:90254). This research was supported by project PID2024-160789OB-I00 funded by MICIU/AEI/10.13039/501100011033/FEDER, UE.

Data availability

Data references are provided in the manuscript.

References

Alam, J., Beneš, R., Beszédeš, M., Burget, L., Dahmane, M., Fathan, A., Ghodrati, H., Glembek, O., Kang, W.H., Matějka, P., Mošner, L., Plchot, O., Rohdin, J., Silnova, A., Stafylakis, T., 2022. Development of ABC systems for the 2021 edition of NIST speaker recognition evaluation. In: The Speaker and Language Recognition Workshop. Odyssey 2022, pp. 346–353.

Alam, J., Boulianne, G., Burget, L., Dahmane, M., Diez Sánchez, M., Lozano-Diez, A., Glembek, O., St-Charles, P.-L., Lalonde, M., Matejka, P., Mizera, P., Monteiro, J., Mosner, L., Noiseux, C., Novotný, O., Plchot, O., Rohdin, J., Silnova, A., Slavicek, J., Stafylakis, T., Wang, S., Zeinali, H., 2020. Analysis of ABC submission to NIST SRE 2019 CMN and VAST challenge. In: The Speaker and Language Recognition Workshop. Odyssey 2020, pp. 289–295.

Barahona, S., Silnova, A., Mošner, L., Peng, J., Plchot, O., Rohdin, J., Zhang, L., Han, J., Palka, P., Landini, F., Burget, L., Stafylakis, T., Cumani, S., Boboš, D., Hlavaček, M., Kodovsky, M., Pavliček, T., 2025. Analysis of ABC frontend audio systems for the NIST-SRE24. In: Interspeech 2025. pp. 5763–5767.

Brümmer, N., 2010. Measuring, Refining and Calibrating Speaker and Language Information Extracted from Speech (Ph.D. thesis). Stellenbosch University, South Africa.

Brümmer, N., du Preez, J.A., 2006. Application-independent evaluation of speaker detection. Comput. Speech Lang. 20 (2–3), 230–275.

Brümmer, N., Silnova, A., Burget, L., Stafylakis, T., 2018. Gaussian meta-embeddings for efficient scoring of a heavy-tailed PLDA model. In: The Speaker and Language Recognition Workshop. Odyssey 2018, pp. 349–356.

Brümmer, N., Swart, A., Mosner, L., Silnova, A., Plchot, O., Stafylakis, T., Burget, L., 2022. Probabilistic spherical discriminant analysis: An alternative to PLDA for length-normalized embeddings. In: Interspeech 2022. pp. 1446–1450.

Burget, L., Plchot, O., Cumani, S., Glembek, O., Matějka, P., Brümmer, N., 2011. Discriminatively trained probabilistic linear discriminant analysis for speaker verification. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 4832–4835.

Chung, J.S., Nagrani, A., Zisserman, A., 2018. VoxCeleb2: Deep speaker recognition. In: Interspeech 2018. pp. 1086–1090.

Colibro, D., Vair, C., Dalmasso, E., Farrell, K., Karvitsky, G., Cumani, S., Laface, P., 2017. Nuance - Politecnico di Torino's 2016 NIST speaker recognition evaluation system. In: Interspeech 2017. pp. 1338–1342.

Cumani, S., Batzu, P.D., Colibro, D., Vair, C., Laface, P., Vasilakakis, V., 2011. Comparison of speaker recognition approaches for real applications. In: Interspeech 2011. pp. 2365–2368.

Cumani, S., Brümmer, N., Burget, L., Laface, P., Plchot, O., Vasilakakis, V., 2013. Pairwise discriminative speaker verification in the I-Vector space. IEEE/ACM Trans. Audio Speech Lang. Process. 21 (6), 1217–1227.

Cumani, S., Glembek, O., Brümmer, N., Villiers, E., Laface, P., 2012. Gender independent discriminative speaker recognition in i-vector space. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 4361–4364.

Cumani, S., Laface, P., 2014a. Generative pairwise models for speaker recognition. In: The Speaker and Language Recognition Workshop. Odyssey 2014, pp. 273–279.

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

Cumani, S., Laface, P., 2014c. Training pairwise support vector machines with large scale datasets. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 1645–1649.

Cumani, S., Laface, P., 2018. Speaker recognition using e–vectors. IEEE/ACM Trans. Audio Speech Lang. Process. 26 (4), 736–748.

Cumani, S., Laface, P., 2019. Exact memory–constrained UPGMA for large scale speaker clustering. Pattern Recognit. 95, 235–246.

Cumani, S., Sarni, S., 2022. Impostor score statistics as quality measures for the calibration of speaker verification systems. In: The Speaker and Language Recognition Workshop. Odyssey 2022, pp. 25–32.

Cumani, S., Sarni, S., 2023a. The distributions of uncalibrated speaker verification scores: A generative model for domain mismatch and trial-dependent calibration. IEEE/ACM Trans. Audio Speech Lang. Process. 31, 2204–2219.

Cumani, S., Sarni, S., 2023b. From adaptive score normalization to adaptive data normalization for speaker verification systems. In: Interspeech 2023. pp. 5296–5300.

Cumani, S., Silnova, A., Barahona, S., Mošner, L., Plchot, O., Rohdin, J., 2025. Analysis of the ABC classification backends for NIST SRE24. In: Interspeech 2025. pp. 3978–3982.

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

Deng, J., Guo, J., Xue, N., Zafeiriou, S., 2019. Arcface: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4690–4699.

Desplanques, B., Thienpondt, J., Demuynck, K., 2020. ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In: Interspeech 2020. pp. 3830–3834.

Gao, S.-H., Cheng, M.-M., Zhao, K., Zhang, X.-Y., Yang, M.-H., Torr, P., 2019. Res2Net: A new multi-scale backbone architecture. IEEE Trans. Pattern Anal. Mach. Intell.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 770–778.

Huh, J., Chung, J.S., Nagrani, A., Brown, A., Jung, J.-w., Garcia-Romero, D., Zisserman, A., 2024. The VoxCeleb speaker recognition challenge: A retrospective. IEEE/ACM Trans. Audio Speech Lang. Process. 32, 3850–3866.

Ioffe, S., 2006. Probabilistic linear discriminant analysis. In: Proceedings of the 9th European Conference on Computer Vision. ECCV '06, Vol. Part IV, pp. 531–542.

Karam, Z., Campbell, W., Dehak, N., 2011. Towards reduced false-alarms using cohorts. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 4512–4515.

Kenny, P., 2010. Bayesian speaker verification with heavy-tailed priors. In: The Speaker and Language Recognition Workshop. Odyssey 2010.

Li, L., Li, X., Jiang, H., Chen, C., Hou, R., Wang, D., 2023. CN-celeb-AV: A multi-genre audio-visual dataset for person recognition. In: Interspeech 2023. pp. 2118–2122.

Lin, Y., Cheng, M., Zhang, F., Gao, Y., Zhang, S., Li, M., 2024. VoxBlink2: A 100K+ speaker recognition corpus and the open-set speaker-identification benchmark. In: Interspeech 2024. pp. 4263–4267.

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

Matějka, P., Novotný, O., Plchot, O., Burget, L., Sánchez, M.D., Černocký, J., 2017. Analysis of score normalization in multilingual speaker recognition. In: Interspeech 2017. pp. 1567–1571.

McLaren, M., Ferrer, L., Castan, D., Lawson, A., 2016. The speakers in the wild (SITW) speaker recognition database. In: Interspeech 2016. pp. 818–822.

Nagrani, A., Chung, J.S., Xie, W., Zisserman, A., 2019. Voxceleb: Large-scale speaker verification in the wild. Comput. Sci. Lang.

2018. NIST 2018 speaker recognition evaluation plan. URL https://www.nist.gov/itl/iad/mig/nist-2018-speaker-recognition-evaluation.

2019. NIST 2018 speaker recognition evaluation plan. URL https://www.nist.gov/itl/iad/mig/nist-2019-speaker-recognition-evaluation.

2021. NIST 2021 speaker recognition evaluation plan. URL https://www.nist.gov/itl/iad/mig/nist-2021-speaker-recognition-evaluation-sre21.

2024. NIST 2024 speaker recognition evaluation plan. URL https://www.nist.gov/itl/iad/mig/nist-2024-speaker-recognition-evaluation-sre24.

Okabe, K., Koshinaka, T., Shinoda, K., 2018. Attentive statistics pooling for deep speaker embedding. In: Interspeech 2018. pp. 2252–2256.
Povey, D., Cheng, G., Wang, Y., Li, K., Xu, H., Yarmohammadi, M., Khudanpur, S., 2018. Semi-orthogonal low-rank matrix factorization for deep neural networks. In: Interspeech 2018. pp. 3743–3747.
Prince, S.J.D., Elder, J.H., 2007. Probabilistic linear discriminant analysis for inferences about identity. In: Proceedings of 11th International Conference on Computer Vision. pp. 1–8.
Sadjadi, O., 2021. NIST SRE CTS superset: A large-scale dataset for telephony speaker recognition. URL https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=933116.
Sadjadi, S.O., Greenberg, C., Singer, E., Mason, L., Reynolds, D., 2022. The 2021 NIST speaker recognition evaluation. In: The Speaker and Language Recognition Workshop. Odyssey 2022, pp. 322–329.
Sholokhov, A., Kuzmin, N., Lee, K.A., Chng, E.S., 2023. Probabilistic back-ends for online speaker recognition and clustering. In: 2023 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 1–5.
Silnova, A., Brümmer, N., Garcia-Romero, D., Snyder, D., Burget, L., 2018. Fast variational Bayes for heavy-tailed PLDA applied to i-vectors and x-vectors. In: Interspeech 2018. pp. 72–76.
Silnova, A., Brümmer, N., Swart, A., Burget, L., 2023. Toroidal probabilistic spherical discriminant analysis. In: 2023 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 1–5.
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, pp. 5329–5333.
Sturim, D., Reynolds, D.A., 2005. Speaker adaptive cohort selection for tnorm in text-independent speaker recognition. In: 2005 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, Vol. 1, pp. I/741–I/744, Vol. 1.
Thienpondt, J., Demuynck, K., 2023. ECAPA2: A hybrid neural network architecture and training strategy for robust speaker embeddings. In: 2023 IEEE Automatic Speech Recognition and Understanding Workshop. ASRU, pp. 1–8.
Van Leeuwen, D., Brümmer, N., 2007. An introduction to application-independent evaluation of speaker recognition systems. Lecture Notes in Comput. Sci. 4343, 330–353.
Villalba, J., Chen, N., Snyder, D., Garcia-Romero, D., McCree, A., Sell, G., Borgstrom, J., García-Perera, L.P., Richardson, F., Dehak, R., Torres-Carrasquillo, P.A., Dehak, N., 2020. State-of-the-art speaker recognition with neural network embeddings in NIST SRE18 and speakers in the wild evaluations. Comput. Speech Lang. 60, 101026.
Wang, S., Chen, Z., Han, B., Wang, H., Liang, C., Zhang, B., Xiang, X., Ding, W., Rohdin, J., Silnova, A., et al., 2024. Advancing speaker embedding learning: Wespeaker toolkit for research and production. Speech Commun. 162, 103104.
Wang, Q., Lee, K.A., Liu, T., 2022. Scoring of large-margin embeddings for speaker verification: Cosine or PLDA? In: Interspeech 2022. pp. 600–604.
Wang, H., Liang, C., Wang, S., Chen, Z., Zhang, B., Xiang, X., Deng, Y., Qian, Y., 2023. Wespeaker: A research and production oriented speaker embedding learning toolkit. In: IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 1–5.
Yakovlev, I., Makarov, R., Balykin, A., Malov, P., Okhotnikov, A., Torgashov, N., 2024. Reshape dimensions network for speaker recognition. In: Interspeech 2024. pp. 3235–3239.
Yang, L., Zhang, R.-Y., Li, L., Xie, X., 2021. SimAM: A simple, parameter-free attention module for convolutional neural networks. In: Meila, M., Zhang, T. (Eds.), Proceedings of the 38th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 139, PMLR, pp. 11863–11874.
Zadrozny, B., Elkan, C., 2002. Transforming classifier scores into accurate multiclass probability estimates. Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.