The Speaker and Language Recognition Workshop (Odyssey 2022)
28 June - 1 July 2022, Beijing, China

Analyzing speaker verification embedding extractors and back-ends under language and channel mismatch

Anna Silnova[1], *Themos Stafylakis*[2], *Ladislav Mošner*[1], *Oldřich Plchot*[1], *Johan Rohdin*[1], *Pavel Matějka*[1], *Lukáš Burget*[1], *Ondřej Glembek*[1], *Niko Brummer*[3]

[1]Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Brno, Czechia
[2]Omilia - Conversational Intelligence, Athens, Greece
[3]Phonexia, South Africa

Abstract

In this paper, we analyze the behavior and performance of speaker embeddings and the back-end scoring model under domain and language mismatch. We present our findings regarding ResNet-based speaker embedding architectures and show that reduced temporal stride yields improved performance. We then consider a PLDA back-end and show how a combination of small speaker subspace, language-dependent PLDA mixture, and nuisance-attribute projection can have a drastic impact on the performance of the system. Besides, we present an efficient way of scoring and fusing class posterior logit vectors recently shown to perform well on speaker verification task. The experiments are performed using the NIST SRE 2021 setup.

1. Introduction

In this paper, we provide an analysis of an integral part of state-of-the-art speaker recognition systems, which is a DNN-based embedding extractor and a subsequent speaker back-end that provides probabilistic verification scores for individual trials. We focus on the most recent NIST SRE 2021 which provided us with an opportunity for experimentation and this analysis.

The past line of speaker recognition (SR) research focused on modeling the fixed-length utterance representations, such as i-vectors [1] obtained as maximum a-posteriori estimates of a latent variable in generative factor analysis model. The obtained i-vector representations (generative embeddings) were subsequently modeled by probabilistic linear discriminant analysis (PLDA) [2], a technique introduced in face verification.

Deep neural networks (DNNs) have been gradually incorporated into the speaker recognition pipeline, through replacing or improving one or more of the components of an i-vector + PLDA system (e.g. feature extraction, calculation of sufficient statistics, i-vector extraction or PLDA classifier). For instance on the front-end level, employment of DNN bottleneck features (BNF) instead of conventional MFCC features [3], or simply concatenating BNF and MFCCs [4] and more recently learning the feature extraction directly from the raw audio [5, 6, 7], was proposed. Later in the modeling stage, NN acoustic models were proposed to replace generative Gaussian mixture models (GMM) for extraction of sufficient statistics [8, 9], or for complementing or substituting the PLDA [10, 11], [12], respectively.

New deep learning works have logically resulted in attempts to train a larger DNN directly for speaker recognition tasks, i.e., binary classification of two utterances as a *target* or a *non-target* trial [13, 14, 15, 16]. Such systems are denoted as *end-to-end* systems and were proven competitive for text-dependent tasks [13, 14, 17] as well as for text-independent tasks considering short test utterances and an abundance of training data [15]. On text-independent tasks with longer utterances and moderate amount of training data, the i-vector motivated end-to-end system [16] outperformed generative baselines, but at the cost of high complexity during training.

While the fully end-to-end SR systems have been struggling with large requirements on the amount of training data (often not available to the researchers) and high computational costs, the focus on speaker recognition has partially shifted back to generative modeling, but now with utterance representations obtained from a single DNN. Such DNN takes the frame-level features of an utterance as an input and directly produces an utterance-level representation, usually referred to as an *embedding* [18, 13, 14, 19, 20, 21]. The embedding is obtained by the means of a *pooling mechanism* (for example taking the mean) over the frame-wise outputs of one or more layers in the DNN [18], or by the use of a recurrent DNN [13]. One effective approach is to train the DNN for classifying a set of training speakers, i.e., using multi-class training [18, 19, 20]. In order to perform speaker verification, the embeddings are extracted and used in a standard back-end, e.g. PLDA [19, 20].

Most recently, deep convolutional neural networks (DCNNs) models such as ResNet [22] play the core role in the SR systems and are continuously replacing feed forward DNNs [20] for embedding extraction. DCNNs are often fine-tuned via optimizing angular margin loss as in face verification. This approach offered the best results in the domain of English wide-band data [23] and has been competitive in more challenging and less data-rich domain of telephone Tunisian Arabic data [24] as well as in the most systems submitted to the recent NIST SRE 2020 and 2021 challenges which always showcase the latest state-of-the-art modeling techniques.

Great deal of our analysis will explore subtle changes in the standard ResNet architecture such as changing the temporal strides or different pooling mechanisms. We will compare the performance of operationally practical and relatively small ResNet34 with a much larger ResNet152 and we will provide an analysis with fine-tuning these models for longer duration segments often present during inference as these are typically in sharp contrast with very short (and same length) segments used during training.

As it is typical in NIST SREs, also the last SRE21 brought a challenging scenario of a channel, language, and data mismatch w.r.t. training data. We are therefore exploring an approach to incorporate language information into PLDA framework, vari-

10.21437/Odyssey.2022-2

ous nuisance variability suppression techniques, and score normalization. These methods, in our experiments, led to better results than using a simple cosine-distance scoring. We also experiment with taking the embeddings from different parts of the DNN and use our back-ends to model directly the output of the pooling layer, standard (x-vector) embedding, and also we provide an insight into scoring with class posterior logits from the very end of the ResNet embedding extractor.

2. Embedding extractors and Back-end model

In this section, we present the basic architecture of the embedding extractors used in the experimental part of this paper. Also, we describe in detail the back-end approach used in the majority of experiments. When presenting the experiments we will always describe the differences in the setup with respect to the models and approaches described here.

2.1. ResNet Architectures

Residual Networks (ResNets), together with their variants (e.g. ResNeXt, Res2Net [25]) are standard choices in speaker recognition research. In this paper, we examine two different standard ResNet architectures, namely ResNet34 and ResNet152. ResNet34 provides a good compromise between accuracy and computational efficiency, e.g. it can be deployed in CPU-based production systems. On the other hand, ResNet152 is a very deep architecture that can yield state-of-the-art results but requires GPU in runtime for being deployed in production systems that perform real-time processing. In terms of architectural investigation, we experiment with the following directions.

2.1.1. Statistics pooling

Apart from the typical mean and standard deviation (std) statistics pooling, we examine using only standard deviation features in the statistics pooling layer. The approach was examined in [26] and appears to generalize better, at least in cases of dataset-shifts between training and test. Furthermore, we experimented with the recently proposed correlation pooling, which showed good improvements in VoxCeleb [27].

2.1.2. Temporal stride

We also experiment with reducing the temporal stride of the ResNet blocks. The temporal stride per ResNet stage is set to [1,2,1,2] or [1,1,2,1] (i.e. a cumulative stride equal to 4 or 2 instead of the standard 8) while the frequency stride is the typically used [1,2,2,2]. The motivation is to reduce the receptive field in order to model shorter speech patterns, which should be more language-independent. We should mention though that reducing the stride results in increased computational requirements, and hence the model becomes less efficient.

2.1.3. Extracting alternative speaker representations

Another set of experiments is related to the layer from which the speaker representation is extracted. One alternative to the standard embeddings is extracting the statistics. Our motivation is that embeddings are susceptible to overfitting the training speakers and languages, as they interact directly with the classification head. Consider for example TDNN-based x-vectors, where two low-dimensional representations can serve as embeddings. The experiments show that embeddings extracted from the input to the classification head are inferior [28]. On the other hand, in ResNet-based architectures there is a single low-dimensional representation after statistics pooling. We experiment with modifying the architecture by adding a second hidden layer however the results were discouraging. We, therefore, consider to extract the statistics instead. Statistics allows us to experiment with unsupervised dimensionality reduction methods (e.g. PCA), use a training set that is more diverge compared to that of the extractor, and possibly retain directions that are more discriminative for languages and domains not included in the training set.

Apart from using the statistics, we examine the recently proposed method of extracting the logits, i.e. the projection of the embeddings to the classification head (prior to the softmax function). The method was proposed in [29] and yields state-of-the-art results in SRE21. We moreover show that the method can be implemented more efficiently using Cholesky (or other equivalent) decomposition of symmetric positive (semi-)definite matrices.

2.1.4. Fine-tuning on long durations

Several papers have reported improvements by fine-tuning the extractor on long durations for the last few epochs. However, most of the results are based on VoxCeleb and use cosine similarity scoring. It is therefore interesting to examine whether similar improvements will be attained on more challenging setups and with a PLDA back-end.

2.2. Back-end Architectures

In most of the experiments described in this paper, we follow the same approach to pre-processing the embeddings and training the back-end:

2.2.1. Nuisance attribute projection

We start by removing from the data the direction corresponding to speaker gender by nuisance attribute projection (NAP) [30, 31]. Further on in Section 3.9 we present the experiment motivating for suppressing gender information. Then, we proceed with centering the data, LDA reducing dimensionality of the embeddings to 100 (see Section 3.7 for details), and length normalization.

2.2.2. Mixture of language-dependent PLDAs

After data pre-processing, we train a mixture of 3 PLDA models [32, 33]: each component of the mixture is a PLDA trained on the data coming from one of three languages: English, Cantonese, and Mandarin. At test time, we score each trial with each language-dependent PLDA model, the final score of the mixture is a weighted average of the scores of individual mixture components:

$$s_{e,t} = \sum_{p \in \{eng, cmn, yue\}} w^p_{e,t} s^p_{e,t}, \tag{1}$$

where $s^p_{e,t}$ are LLR scores computed with one of the mixture components (PLDA model M_p) and $w^p_{e,t}$ are corresponding weights given by:

$$w^p_{e,t} = \frac{1}{2}\left(P(\mathcal{R}_e \mid M_p) + P(\mathcal{R}_t \mid M_p)\right). \tag{2}$$

In the expression above, $\mathcal{R}_e$ and $\mathcal{R}_t$ are enrollment and test embeddings (one or three in case of multi-session enrollment models). Likelihoods $P(\mathcal{R} \mid M_p)$ can be estimated using e.g. equation (19) of [34].

Notice, that our approach to estimating the score is somewhat simpler than that of [32], where the correct LLR of the mixture was estimated. At early stage of the development, we compared these two approaches but observed a slight performance degradation when using more complicated by-the-book scoring with the mixture of PLDA models and decided to proceed with the easier approach described above.

3. Experiments and results

3.1. Experimental setup

For training our embedding extractors we used three databases, namely NIST CTS Superset [35], Voxceleb 1 and 2 [36, 37]. In all our experiments, the systems are trained on 8kHz data, all 16kHz data from the aforementioned datasets are downsampled to 8kHz. In total, there are 14096 training speakers. We used Kaldi style augmentation with MUSAN database [38]. At the input of the embedding extractors there are 64 mel filter banks with frequency band limited to 20-3800Hz.

When training the back-end model, we use only NIST CTS Superset (English, Cantonese, and Mandarin recordings from this set). We use the embeddings extracted from the original recordings along with one augmented copy of the data. Each recording is augmented with one augmentation that was randomly selected out of five types of augmentation used when training the extractor. There are approximately 7000 speakers used for training the back-end.

In all of our experiments, we test the performance on NIST SRE 21 evaluation set. The particulars of the evaluation can be found in the evaluation plan [39]. The relevant details for our system design choices are: there are no cross-gender trials; the majority of the utterances are in English, Cantonese, and Mandarin; there are cross-language trials. We report the results in terms of Equal Error Rate (EER) and minimum cost (min_C) as computed by the scoring tool released by NIST as a part of the evaluation.

Below, we present the results of the experiments where we vary different aspects of the embedding extractors described in Sections A.1 and A.2 and the back-end model of Section 2.2. We start with the experiments on embedding extractors and then continue with the back-end related experiments.

3.2. Effect of temporal stride

As we know, the evaluation set mostly consists of non-English data, while the majority of the recordings used for training the embedding extractors are in English. Hence, there is a need to compensate for the language mismatch between training and evaluation data. Our assumption is that by reducing the temporal context, we encourage the network to train on more language-independent patterns. To verify this assumption, we trained several ResNet networks with three settings for temporal stride: strides were set to [1,2,2,2], [1,2,1,2], or [1,1,2,1] per ResNet stage (i.e. a cumulative stride equals to 8, 4 and 2, respectively), frequency stride in all cases was fixed to [1,2,2,2]. We experimented with two different pooling mechanisms (more information is given in Section 3.3). The back-end model for all experiments was as described in 2.2.

The results of the extractors with varying temporal strides are shown in Table 1. As one might notice, reducing the cumulative temporal stride from 8 to 4 (compare lines 1 and 2 of the table) results in a noticeable performance improvement for both pooling mechanisms in case of ResNet34. For the larger embedding extractor, the performance improvement is not as pronounced but still non-negligible. However, reducing the stride even further degrades the performance (we have the results only for ResNet34 with std pooling) indicating that for successful performance the network has to observe at least some temporal dependencies in the data.

3.3. Statistics pooling

Here, we compare three approaches to pool frame-level representations in the embedding extractor. The first option we consider is the standard way of statistics pooling when both mean and std are computed. Second, we consider the case when only standard deviation features are used for statistics pooling. In [26], this approach was shown to generalize better when there is a mismatch between training and test data. Finally, we use the approach of [27] where correlation pooling is used instead of mean and std.

We compare these three approaches by training ResNet34 embedding extractors with the aforementioned pooling mechanisms. The results are presented in Table 2. As can be seen, both mean+std and std alone result in a similarly good performance in terms of min_C, however, in terms of EER, using just standard deviation brings considerable gain. Regarding correlation pooling, we notice that there is a notable performance degradation compared to the other two options.

3.4. Extracting statistics as speaker representations

Typically, speaker embeddings are the activations of the low-dimensional penultimate layer of the extractor network. Hence, the embeddings extracted in this way might be overtuned towards training speakers and domains. To overcome this potential problem, we try to extract the speaker representation directly as the output of the statistics pooling layer. As this layer is further away from the classification head and has a higher dimension, our expectation is that such a representation would be less susceptible to overtraining to the speakers that were used when training the network. However, because such representations have relatively high dimensionality (2048 in our case), it is problematic to directly use them for training the back-end model. For this reason, we need to perform dimensionality reduction of high-dimensional statistics first. We believe that if dimensionality reduction is trained on the set of speakers other than that was used for training the network then, even though the resulting embeddings are low-dimensional vectors, they will be more suitable for a general speaker recognition problems than the original embeddings.

In our experiments, we tried the simplest option for dimensionality reduction: Principal Component Analysis (PCA). PCA projection matrix is estimated on the back-end training set and reduces the dimensionality of statistics from 2048 to 256 (size of the original embeddings).

3.5. Extracting logits as speaker representations

In the opposite direction, the authors in [29] propose to use high-dimensional vectors of class posterior logits (cl-embeddings) in place of conventional embeddings. They show performance improvements when such vectors are used in combination with cosine distance scoring. However, we want to point out that using high-dimensional vectors of logits is not necessary. Instead, one can compute a low-dimensional projection of the original embeddings such that scoring high-dimensional cl-embeddings will be equivalent to scoring low-

Table 1: Comparison of speaker verification performance of ResNet embedding extractors with different temporal strides (two different pooling approaches for ResNet34). The back-end used is a mixture of 3 language-dependent PLDA models.

		ResNet34				**ResNet152**	
	Temp stride	**std**		**corr**			
		min_C	**EER (%)**	**min_C**	**EER (%)**	**min_C**	**EER (%)**
1	8:[1,2,2,2]	0.495	8.66	0.553	9.84	0.412	6.32
2	4:[1,2,1,2]	0.473	8.38	0.520	9.44	0.407	6.28
3	2:[1,1,2,1]	0.504	9.07	-	-	-	-

Table 2: Comparison of performance of embedding extractors utilizing different pooling strategies. In all cases, the architecture we use is ResNet34 with cumulative temporal stride 4. The back-end used is a mixture of 3 language-dependent PLDA models.

	Pooling	**min_C**	**EER (%)**
1	mean+std	0.474	8.74
2	std	0.473	8.38
3	correlation	0.520	9.44

dimensional projected embeddings. Below, we demonstrate it for cosine scoring, but a similar approach can be used for training a PLDA model.

When scoring cl-embeddings, we make use of the fact that the logits $\mathbf{l}$ are just high-dimensional projection of the original embeddings $\mathbf{r}$:

$$\mathbf{l} = \mathbf{K}\mathbf{r}, \tag{3}$$

where $\mathbf{K}$ is the projection matrix of shape $N_s \times d$ (number of training speakers and embedding dimension, respectively). Then the cosine score between enrolment and test becomes:

$$s_{e,t} = \frac{\mathbf{l}_e'\mathbf{l}_t}{\|\mathbf{l}_e\|\,\|\mathbf{l}_t\|} = \frac{\mathbf{r}_e'\mathbf{K}'\mathbf{K}\mathbf{r}_t}{\|\mathbf{K}\mathbf{r}_e\|\,\|\mathbf{K}\mathbf{r}_t\|}. \tag{4}$$

The matrix $\mathbf{K}'\mathbf{K}$ is a symmetric positive (semi-)definite matrix and can be considered as N_s times the between-speaker covariance, estimated with the speakers contained in the training set of the extractor (recall that when AAM loss is used, the rows of $\mathbf{K}$ have unit norm). By performing Cholesky decomposition we obtain the upper triangular $d \times d$ matrix $\mathbf{M}$ such that $\mathbf{K}'\mathbf{K} = \mathbf{M}'\mathbf{M}$. As a result:

$$s_{e,t} = \frac{\mathbf{r}_e'\mathbf{M}'\mathbf{M}\mathbf{r}_t}{\|\mathbf{M}\mathbf{r}_e\|\,\|\mathbf{M}\mathbf{r}_t\|}. \tag{5}$$

Thus, instead of extracting and scoring with high-dimensional logit vectors $\mathbf{l} = \mathbf{K}\mathbf{r}$, we may equivalently extract low-dimensional vectors $\mathbf{M}\mathbf{r}$. Finally, an efficient implementation of the fusion method of logits (proposed also in [29]) is derived in Appendix B.

We perform the experiment where we compare three approaches: conventional embeddings (line 1 of Table 3), outputs of the statistics pooling layer in combination with PCA (line 2), and embeddings projected with low-dimensional square matrix $\mathbf{M}$, computed as described above (line 3). For all three types of speaker embeddings, we use either cosine scoring or train the mixture of language-dependent PLDA models as described in Section 2.2. In the latter case, $\mathbf{r}$ are projected onto $\mathbf{M}$ and length-normalized, prior to LDA and a new length-normalization. Here, we perform the experiments only with ResNet34 embedding extractor with cumulative temporal stride set to 4 and standard deviation pooling. The results of this experiment are presented in Table 3.

In case of cosine scoring, we observe major performance gains from using cl-embeddings compared to conventional embedding vectors, which agrees with the findings of [29]. Using statistics in this scenario, however, results in severe degradation. For the second back-end approach, we do not observe such high variability in the results. Statistics, though, still provide inferior results compared to the other two types of speaker representations. One of the possible explanations is the dimensionality reduction approach that we use. In future, we plan to investigate alternatives to PCA.

Table 3: Comparison of performance of three types of embeddings extracted from ResNet34 with temporal stride 4 and std pooling mechanism.

		cos		**mix PLDA**	
		min_C	**EER (%)**	**min_C**	**EER (%)**
1	embd	0.662	15.60	0.473	8.38
2	stats+PCA	0.911	27.57	0.527	9.51
3	cl-embd	0.612	12.22	0.475	8.35

As both types of embedding extractors that we use here were trained on exactly the same dataset and the main difference between them is only the size of the network, when fusing two systems (either using score level fusion or the approach described in appendix B) we do not observe any performance improvements compared to using just the bigger ResNet152 model. Thus, we do not report any fusion results here.

3.6. Fine-tuning the extractor on long segments

Previously (see e.g. [40]) and in system descriptions of several top performing teams of NIST SRE 2021 (e.g. in [29]), it was noted that fine-tuning (FT) a pretrained embedding extractor on long segments (10-20s vs. 2-4s) results in a major performance gain. In order to verify this finding, we perform the following experiment: we fine-tune our two embedding extractors on 10s long segments. The embeddings extracted from the fine-tuned models are scored with the approach described in Section 2.2.

The results of this experiment are presented in Table 4. It is seen that for both extractors fine-tuning procedure provides a considerable performance improvement. Also, we can note that the effect of the fine-tuning does not depend on the back-end approach: in previous works, either cosine scoring or a single PLDA model were used, while we observe similar performance improvements with a mixture of PLDA models.

3.7. Back-end processing

In this section, we investigate the impact of individual steps of the back-end recipe described in Section 2.2 compared to a standard back-end approach: a single PLDA model trained on the embeddings that were centered, their dimensionality reduced

Table 4: Effect of fine-tuning embedding extractors on 10s long segments on speaker verification performance. Two types of networks are used: ResNet34 with std pooling and ResNet152 with mean and std pooling. For both networks, temporal stride is set to 4. The back-end in all cases is a mixture of 3 language-dependent PLDA models.

	FT	ResNet34		ResNet152	
		min_C	EER (%)	min_C	EER (%)
1	no	0.473	8.38	0.407	6.28
2	yes	0.432	7.38	0.380	5.70

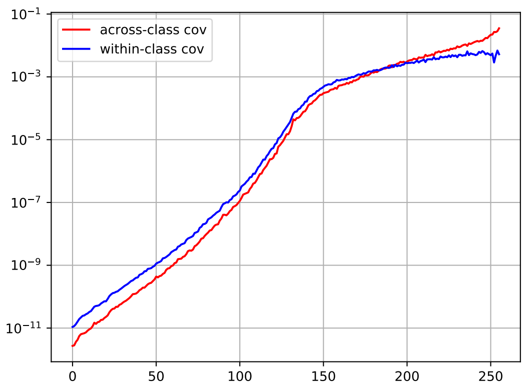

Figure 1: Diagonal across- and within-class covariances of the back-end training data. Prior to plotting the graphs, the embeddings were projected onto eigenvectors of the total covariance matrix i.e. total covariance is diagonal with elements sorted by the index on horizontal axis.

from 256 to 200 by LDA and then length-normalized. Table 5 presents gradual improvements in the verification performance with step-by-step transformation of the baseline back-end into the back-end described in Section 2.2. The same experiments were performed for embeddings extracted with ResNet34 and ResNet152 models with temporal strides set to [1,2,1,2].

From the results, the largest improvement was observed when reducing the dimensionality of the embedding after LDA from 200 to 100 (lines 1 and 2 of Table 5). Figure 1 gives an insight into this phenomenon. It displays eigenvalues of the across- and within-class covariance matrices estimated on the PLDA training data. The eigenvalues were computed after projecting the embeddings with eigenvectors of the total covariance matrix i.e. the elements of total covariance monotonically increase from left to right. Figure 1 shows the plots for ResNet34 embedding extractor, similar pattern is observed for ResNet152. By inspecting the graphs, we notice that roughly half of the embedding dimensions have very little variability. When such directions are selected by LDA, their (possibly noisy) variability is amplified leading to performance degradation.

Introducing a mixture of language-dependent PLDA models instead of a single one (lines 2 and 3 of the table) brings another considerable improvement. Finally, removing the direction corresponding to the highest gender variability with NAP results in an additional performance gain. However, compared to the previous two modifications the improvement from using NAP is rather small.

3.8. Alternative back-end approaches

Here, we want to compare the back-end strategy presented in Section 2.2 with some other approaches that were shown to be effective in NIST SRE21. Namely, we have noticed that many teams used cosine similarity scoring instead of PLDA back-end; in [29], it was shown that it is beneficial to use score and channel normalization when doing cosine similarity scoring of the embeddings. Also, as mentioned in Section 3.5, cl-embeddings (logits from the embedding extractor network) in combination with cosine scoring were shown to provide competitive speaker verification results. In order to show the effectiveness of the mentioned approaches we use three scoring strategies: cosine scoring on the embeddings and on the cl-embeddings and our approach with NAP, LDA and a mixture of 3 PLDA models. For each of these strategies, we report the results with and without score, channel, and combination of both normalization.

The cohort for score normalization is constructed from NIST CTS Superset by averaging the embeddings on per-speaker basis i.e. the cohort contains one embedding per speaker from the PLDA training set resulting in approximately 7k embeddings. We use 400 highest scores of the enrollment and test segments against the cohort for score normalization.

Channel normalization is a calibration-like technique that shifts and scales evaluation scores based on the channel type of the enrollment-test pair. For each of four types of trials (mic-mic, tel-tel, mic-tel, tel-mic), we evaluate the mean and standard deviation of the scores from NIST SRE21 development set. These parameters are then used to shift and scale evaluation scores for the matching trial type. If both score and channel normalization are used, we perform score normalization first. When scoring cl-embeddings, rather than computing cosine distance scores directly we follow the approach described in Section 3.5 allowing for efficient score estimation. Note that the channel is assumed given in SRE21. In real systems, one should estimate it using a channel classifier.

We report the results of the aforementioned approaches on ResNet34 and ResNet152 embeddings with and without fine-tuning on the long speech segments. The results are shown in Table 6. Several conclusions could be made looking at the results. First, as expected, we notice that for any kind of back-end or score normalization, the bigger ResNet152 provides lower error rates than the smaller ResNet34. Second, our results indicate that channel normalization (and its combination with score normalization) improves the performance only for cosine distance scoring and for the models that were fine-tuned on long speech segments. For the models without fine-tuning, we observe, that typically channel normalization results in the improvements in terms of EER and performance drop in terms of min_C. Also, we verify the observation of [29] that cosine scoring of cl-embeddings outperforms cosine scoring of conventional low-dimensional embeddings. However, comparing the results of the mixture of PLDA models with the simple cosine scoring of cl-embeddings, we note that the more elaborated approach provides considerably better results for all embedding extractors. Thus, we conclude that even though training separate a back-end model adds additional complexity to the pipeline, it pays off in terms of the system performance.

3.9. Nuisance variability suppression

As was noticed in Section 3.7, using NAP to remove the direction corresponding to the highest gender variability from the data resulted in a slight performance improvement. A reasonable question then is whether the gender dimension was the one

Table 5: Effect of the individual steps used for back-end training. The results are shown for ResNet34 with std pooling and ResNet152 with mean and std pooling. Temporal stride for both extractors is set to 4.

	Back-end	**ResNet34**		**ResNet152**	
		min_C	**EER (%)**	**min_C**	**EER (%)**
1	LDA200, LN, PLDA	0.584	11.37	0.569	9.90
2	LDA100, LN, PLDA	0.515	10.16	0.450	7.49
3	LDA100, LN, 3PLDAs	0.481	8.76	0.419	6.46
4	NAP gender, LDA100, LN, 3PLDAs	0.473	8.38	0.407	6.28

Table 6: Comparison of speaker verification performance of different scoring approaches. Two types of embedding extractors are used (ResNet34 with standard deviation pooling and ResNet152 with mean and std pooling, cumulative temporal stride 4 is used for both networks). For both extractors, we present the results with and without fine-tuning on 10s segments.

Back-end	**S-norm**	**Ch-norm**	**ResNet34**				**ResNet152**			
			no FT		**FT**		**no FT**		**FT**	
			min_C	**EER (%)**	**min_C**	**EER (%)**	**min_C**	**EER (%)**	**min_C**	**EER (%)**
cos	no	no	0.662	15.6	0.554	10.89	0.589	12.22	0.49	9.18
	yes	no	0.65	15.25	0.58	10.52	0.606	11.91	0.525	8.68
	no	yes	0.706	15.35	0.552	10.31	0.62	12.6	0.498	8.77
	yes	yes	0.664	14.89	0.507	9.9	0.615	12.28	0.478	8.51
cos cl-embd	no	no	0.612	12.22	0.534	9.61	0.528	9.76	0.456	7.72
	yes	no	0.612	11.71	0.57	9.12	0.538	9.23	0.478	7.3
	no	yes	0.69	12.42	0.548	8.83	0.574	9.85	0.458	7.39
	yes	yes	0.623	11.52	0.49	8.04	0.546	9.38	0.429	6.83
NAP, LDA, mix 3 PLDA	no	no	0.473	8.38	0.432	7.38	0.407	6.28	0.38	5.7
	yes	no	0.476	8.42	0.426	7.48	0.406	6.27	0.372	5.77
	no	yes	0.592	8.78	0.526	7.49	0.493	6.71	0.433	5.83
	yes	yes	0.543	8.89	0.47	7.56	0.44	6.59	0.38	5.73

to remove or there are other sources of unwanted variability in the embedding distribution that we have to deal with. To analyze this question we performed the following experiment: we train several PLDA models, each of them is trained on the same embeddings. What differs between these models is the pre-processing performed on the embeddings: we either do just centering, LDA and LN, or, prior to centering, we remove a few directions from the data corresponding to various sources of variability. Namely, we remove 2 directions corresponding to the largest PCA eigenvalues i.e. the directions of the highest variability of the data (we expect that these should correspond to gender variability), alternatively, we apply NAP to remove gender, language, or dataset (individual subsets of CTS superset) variability. In all of these experiments, we use a single PLDA to isolate the effect of pre-processing.

The results of this experiment are displayed in Table 7, where lines correspond to the different pre-processing steps described above. The table shows the results for ResNet34 that was or was not fine-tuned on long training segments. As results suggest, when the extractor is not fine-tuned on long segments any kind of variability compensation does not provide any significant improvement in terms of min_C, while removing the dimensions corresponding to either gender or language results in some improvement in terms of EER. However, when the network was fine-tuned, there is a clear performance gain from using gender NAP compared to any other variability compensation approaches. We do not have a clear explanation of this phenomenon, it has to be investigated further.

Table 7: Comparison of various nuisance variability suppression approaches. The embedding extractor is ResNet34 with std pooling and cumulative temporal stride set to 4. FT denotes fine-tuning on 10s speech segments.

	pre-process	**no FT**		**FT**	
		min_C	**EER (%)**	**min_C**	**EER (%)**
1	-	0.515	10.16	0.499	10.02
2	PCA top 2	0.519	10.07	0.504	9.94
3	NAP gender	0.511	9.71	0.463	8.51
4	NAP lang	0.514	9.90	0.493	9.48
5	NAP db	0.517	10.12	0.490	9.58

4. Conclusion

In this paper, we proposed a set of methods to improve speaker recognition performance on a challenging NIST SRE 2021. The setup is characterized by language mismatch between trials and limited or no in-domain data for training or fine-tuning models. Our contributions are related to certain architectural improvements of the extractor (such as the use of reduced temporal stride and the use of alternative pooling methods) as well as to the back-end of the system (such as the use of a PLDA mixture and NAP). We also evaluated our methods against a strong novel approach that uses cosine similarity with score and channel normalization, and logits instead of embeddings, for which we demonstrated a more efficient implementation with embedding dimensional speaker representations. The experiments showed that our proposed approach outperforms the baseline, while it does not make use of any channel (oracle or real) classifier, justifying the use of a more probabilistic back-end in setups similar to SRE21.

5. Acknowledgements

The work was supported by Czech Ministry of Interior project No. VJ01010108 ”ROZKAZ”, Czech National Science Foundation (GACR) project NEUREM3 No. 19-26934X, Czech Ministry of Education, Youth and Sports project no. LTAIN19087 ”Multi-linguality in speech technologies”, European Union’s Horizon 2020 project No. 833635 ROXANNE and Horizon 2020 Marie Sklodowska-Curie grant ESPERANTO, No. 101007666. Computing on IT4I supercomputer was supported by the Czech Ministry of Education, Youth and Sports from the Large Infrastructures for Research, Experimental Development and Innovations project ”e-Infrastructure CZ – LM2018140”.

6. References

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-End Factor Analysis For Speaker Verification,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[2] S. J. D. Prince and J. H. Elder, “Probabilistic linear discriminant analysis for inferences about identity,” in *Proc. International Conference on Computer Vision (ICCV)*, Rio de Janeiro, Brazil, 2007.

[3] A. Lozano-Diez, A. Silnova, P. Matějka, O. Glembek, O. Plchot, J. Pešán, L. Burget, and J. Gonzalez-Rodriguez, “Analysis and Optimization of Bottleneck Features for Speaker Recognition,” in *Proceedings of Odyssey 2016*. 2016, vol. 2016, pp. 352–357, International Speech Communication Association.

[4] P. Matějka, O. Glembek, O. Novotný, O. Plchot, F. Grézl, L. Burget, and J. Černocký, “Analysis Of DNN Approaches To Speaker Identification,” in *Proceedings of ICASSP*. 2016, IEEE Signal Processing Society.

[5] Jee-weon Jung, Hee-Soo Heo, Ju-ho Kim, Hye-jin Shim, and Ha-Jin Yu, “Rawnet: Advanced end-to-end deep neural network using raw waveforms for text-independent speaker verification.,” *Proc. Interspeech 2019*, 2019.

[6] Jee-weon Jung, Seung-bin Kim, Hye-jin Shim, Ju-ho Kim, and Ha-Jin Yu, “Improved rawnet with feature map scaling for text-independent speaker verification using raw waveforms,” *Proc. Interspeech 2020*, pp. 1496–1500, 2020.

[7] Weiwei Lin and Man-Wai Mak, “Wav2spk: A simple dnn architecture for learning speaker embeddings from waveforms,” *Proc. Interspeech 2020*, pp. 3211–3215, 2020.

[8] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, “A novel scheme for speaker recognition using a phonetically-aware deep neural network,” in *Proceedings of ICASSP*, May 2014, pp. 1695–1699.

[9] P Kenny, V Gupta, T Stafylakis, P Ouellet, and J Alam, “Deep neural networks for extracting baum-welch statistics for speaker recognition,” in *Proc. Odyssey*, 2014.

[10] S. Novoselov, T. Pekhovsky, O. Kudashev, V. S. Mendelev, and A. Prudnikov, “Non-linear PLDA for i-vector speaker verification,” in *Proceedings of ICASSP*, Sept 2015, pp. 214–218.

[11] G. Bhattacharya, J. Alam, P. Kenny, and V. Gupta, “Modelling speaker and channel variability using deep neural networks for robust speaker verification,” in *2016 IEEE Spoken Language Technology Workshop, SLT 2016, San Diego, CA, USA, December 13-16*, 2016.

[12] O. Ghahabi and J. Hernando, “Deep belief networks for i-vector based speaker recognition,” in *Proceedings of ICASSP*, May 2014, pp. 1700–1704.

[13] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, “End-to-end text-dependent speaker verification,” in *Proceedings of ICASSP*, March 2016, pp. 5115–5119.

[14] S. X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, “End-to-End attention based text-dependent speaker verification,” in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2016, pp. 171–178.

[15] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, “Deep neural network-based speaker embeddings for end-to-end speaker verification,” in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2016, pp. 165–170.

[16] J. Rohdin, A. Silnova, M. Diez, O. Plchot, P. Matějka, and L. Burget, “End-to-end DNN based speaker recognition inspired by i-vector and PLDA,” in *Proceedings of ICASSP*, 2018.

[17] G. Bhattacharya, J. Alam, T. Stafylakis, and P. Kenny, “Deep neural network based text-dependent speaker recognition: Preliminary results,” in *Proc. Odyssey*, 2016.

[18] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, “Deep neural networks for small footprint text-dependent speaker verification,” in *Proceedings of ICASSP*, May 2014, pp. 4052–4056.

[19] G. Bhattacharya, J. Alam, and P. Kenny, “Deep Speaker Embeddings for Short-Duration Speaker Verification,” in *Interspeech 2017*, 08 2017, pp. 1517–1521.

[20] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur, “Deep Neural Network Embeddings for Text-Independent Speaker Verification,” *Proc. Interspeech 2017*, pp. 999–1003, 2017.

[21] T. Stafylakis, J. Rohdin, O. Plchot, P. Mizera, and booktitle=Proc. Interspeech 2019 pages=2863–2867 year=2019 Burget, L., “Self-supervised speaker embeddings,” .

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] Hossein Zeinali, Shuai Wang, Anna Silnova, Pavel Matějka, and Oldřich Plchot, “But system description to voxceleb speaker recognition challenge 2019,” in *Proceedings of The VoxCeleb Challange Workshop 2019*, 2019, pp. 1–4.

[24] J. Alam, G. Boulianne, O. Glembek, A. Lozano Díez, P. Matějka, P. Mizera, J. Monteiro, L. Mošner, O. Novotný, O. Plchot, J. Rohdin, A. Silnova, J. Slavíček, T.Stafylakis, S. Wang, and H. Zeinali, “ABC NIST SRE 2019 CTS System Description,” in *Proceedings of NIST*. 2019, pp. 1–6, National Institute of Standards and Technology.

[25] Tianyan Zhou, Yong Zhao, and Jian Wu, “Resnext and res2net structures for speaker verification,” in *2021 IEEE*

Spoken Language Technology Workshop (SLT). IEEE, 2021, pp. 301–307.

[26] Shuai Wang, Yexin Yang, Yanmin Qian, and Kai Yu, "Revisiting the statistics pooling layer in deep speaker embedding learning," in *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 2021, pp. 1–5.

[27] T. Stafylakis, J. Rohdin, and L. Burget, "Speaker embeddings by modeling channel-wise correlations," in *Proc. Interspeech 2021*, 2021, pp. 501–505.

[28] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *Proceedings of ICASSP*, 2018, pp. 5329–5333.

[29] A. Avdeeva, A. Gusev, I. Korsunov, A. Kozlov, G. Lavrentyeva, S. Novoselov, T. Pekhovsky, A. Shulipa, A. Vinogradova, V. Volokhov, et al., "STC speaker recognition systems for the NIST SRE 2021," *arXiv preprint arXiv:2111.02298*, 2021.

[30] A. Solomonoff, W.M. Campbell, and I. Boardman, "Advances in channel compensation for svm speaker recognition," in *Proceedings of ICASSP*, 2005, vol. 1, pp. I/629–I/632 Vol. 1.

[31] Hagai Aronowitz, "Inter dataset variability compensation for speaker recognition," in *Proceedings of ICASSP*. IEEE, 2014, pp. 4002–4006.

[32] M. Senoussaoui, P. Kenny, N Brümmer, E. de Villiers, and P. Dumouchel, "Mixture of plda models in i-vector space for gender-independent speaker recognition," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[33] D. Garcia-Romero, X. Zhou, and C. Y. Espy-Wilson, "Multicondition training of gaussian plda models in i-vector space for noise and reverberation robust speaker recognition," in *Proceedings of ICASSP*. IEEE, 2012, pp. 4257–4260.

[34] Niko Brummer, "EM for Simple PLDA," 2010, Technical report.

[35] Omid Sadjadi, "NIST SRE CTS Superset: A large-scale dataset for telephony speaker recognition," 2021-08-16 04:08:00 2021.

[36] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech and Language*, 2019.

[37] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman, "Voxceleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.

[38] David Snyder, Guoguo Chen, and Daniel Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[39] O. Sadjadi, C. Greenberg, E. Singer, L. Mason, and D. Reynolds, "NIST 2021 Speaker Recognition Evaluation Plan, NIST SRE," 2021.

[40] D. Garcia-Romero, G. Sell, and A. Mccree, "Magneto: X-vector magnitude estimation network plus offset for improved speaker recognition," in *Proc. Odyssey 2020 the speaker and language recognition workshop*, 2020.

[41] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

A. Architectural details, optimizers and training strategies

A.1. ResNet34

The network is composed on 34 convolutional layers with residual connections. All convolutional kernels are 3×3, the number of channels is (64,128,256,256) and the first convolutional layer also outputs 64 channels. The number of convolutional layers per block is (3, 4, 6, 3). The input features are 64-dimensional fbanks, extracted from 8kHz audio files, and the training segments contain 350 frames.

The networks are trained using multi-speaker classification and with Additive Angular Margin loss with 30 and 0.3 scale and margin, respectively [41]. As optimizer we use stochastic gradient descent with momentum equal to 0.9. The minibatch size is 256, however to fit it in a single GPU we split the minibatch into 16 "microbatches" of 16 examples each and use gradient accumulation. The initial learning rate is 0.2 which we divide by 2 when the loss does not improve for more than 3000 model updates in the held-out set. When the network is finetuned on long speech segments, we keep the margin of AAM softmax at a value of 0.3. The initial learning rate is set to 0.01 and gradually decreased upon reaching a plateau of cross-validation loss until convergence.

A.2. ResNet152

The ResNet152 embedding extractor comprises 152 convolutional layers. The stages of the network consist of 3, 8, 36, and 3 bottleneck blocks with a pre-activation structure and use 64, 128, 256, and 256 channels, respectively. Per-frame representations are aggregated with a traditional mean and standard deviation pooling. The resulting vectors are projected to 256-dimensional embeddings. The model requires 64-dimensional fbank features and is trained to optimize the AAM loss.

B. Efficient fusion in the logit domain

The fusion method in the logit space of two or more extractors trained with the same set of speakers (suggested in [29]) can also be performed in a low-dimensional space. More specifically, assume two networks with embedding dimension equal to d_1 and d_2. The motivation of [29] is to fuse the two systems using the weighted average logit vectors $\mathbf{l}_f = w_1\mathbf{l}_1 + w_2\mathbf{l}_2$, assuming that the number of training speakers and their order (as encoded in $\mathbf{l}$ via $\mathbf{K}$) is the same in both systems. To derive the cosine similarity in the low-dimensional space ($d_f = d_1 + d_2$) we may define the embedding $\mathbf{r}_f = [w_1\mathbf{r}_1', w_2\mathbf{r}_2']'$ of length d_f. Using linear algebra, one may verify that the cosine similarity between two fused logit vectors $\mathbf{l}_{f,e}$ and $\mathbf{l}_{f,t}$ is identical to the cosine similarity between $\mathbf{M}_f\mathbf{r}_{f,e}$ and $\mathbf{M}_f\mathbf{r}_{f,t}$. $\mathbf{M}_f$ is the upper triangular matrix of the Cholesky decomposition of the between-speaker covariance in the d_f-dimensional space, i.e. $\mathbf{M}_f'\mathbf{M}_f = \mathbf{K}_f'\mathbf{K}_f$, where $\mathbf{K}_f = [\mathbf{K}_1, \mathbf{K}_2]$ is the matrix of concatenated projection matrices of shape $N_s \times d_f$. Therefore, logit-domain fusion can be performed in the d_f-dimensional space, by projecting the concatenated embeddings $\mathbf{r}_f$ onto $\mathbf{M}_f$.