LEAP System for SRE19 CTS Challenge - Improvements and Error Analysis

Shreyas Ramoji, Prashant Krishnan, Bhargavram Mysore, Prachi Singh, Sriram Ganapathy

LEAP Lab, Electrical Engineering, Indian Institute of Science, Bangalore.

shreyasr@iisc.ac.in

Abstract

The NIST Speaker Recognition Evaluation - Conversational Telephone Speech (CTS) challenge 2019 was an open evaluation for the task of speaker verification in challenging conditions. In this paper, we provide a detailed account of the LEAP SRE system submitted to the CTS challenge focusing on the novel components in the back-end system modeling. All the systems used the time-delay neural network (TDNN) based x-vector embeddings. The x-vector system in our SRE19 submission used a large pool of training speakers (about 14k speakers). Following the x-vector extraction, we explored a neural network approach to backend score computation that was optimized for a speaker verification cost. The system combination of generative and neural PLDA models resulted in significant improvements for the SRE evaluation dataset. We also found additional gains for the SRE systems based on score normalization and calibration. Subsequent to the evaluations, we have performed a detailed analysis of the submitted systems. The analysis revealed the incremental gains obtained for different training dataset combinations as well as the modeling methods.

1. Introduction

The recent years have seen increasing demand for authentication and verification systems using speech. In defense applications, speaker detection is an important aspect in surveillance of telephone recordings while in commercial applications like banking, voice-operated smart speakers and mobile phones, the use of speech based authentication is becoming ubiquitous. The acceptable performance of the system relies on relatively clean recordings and with matched languages used in training and testing the systems. The performance is substantially degraded in noisy and multi-lingual environments making the downstream applications vulnerable. Over the past two decades, the NIST speaker recognition evaluation (SRE) challenges provide a suitable benchmark for comparing and standardizing speaker recognition systems. The NIST Speaker Recognition Evaluation 2019 [1] is the latest among an ongoing series of challenges, and it consisted of two tracks - the first was a leaderboard style evaluation on speaker detection from Conversational Telephone Speech (CTS), and the second one was a multimedia speaker recognition. This paper reports the efforts of the LEAP system submission to the SRE19 CTS challenge which advance our previous efforts on the SRE18 challenge [2].

The conventional approach deployed for speaker recognition consisted of the Gaussian mixture modeling (GMM) of speech training data followed by an adaptation using maximum-aposteriori (MAP) principles [3]. The adapted model is compared with the background GMM model using the log-likelihood ratio score. This approach was advanced by the development of i-vectors as fixed dimensional front-end features for speaker recognition tasks [4, 5]. The i-vectors capture long term information of the speech signal such as speaker and language. In the recent past, the i-vectors derived from deep neural network (DNN) based posterior features were also explored for SID [6]. The use of bottleneck features for front-end feature extraction derived from a speech recognition acoustic model has also shown good improvements for speaker recognition [7].

Recently, neural network embedding extractor trained for a supervised speaker discrimination task has shown improvements over the i-vector approach. This uses a time delay neural network (TDNN) with a sequence summary layer followed by feed-forward neural network layers that map to the target layer of training speaker classes. The output of the first feed-forward layer following the sequence summary layer is used as embeddings (x-vectors) for speaker recognition [8]. Following the extraction of x-vectors/i-vectors, different speaker verification systems make use of discriminative/generative models in the back-end for computing the scores. The most popular approaches for scoring include support vector machines (SVMs) [9, 10], Gaussian back-end model [11, 2] and the probabilistic linear discriminant analysis (PLDA) [12, 13]. Some efforts on pairwise generative and discriminative modeling are discussed in [10, 14, 11].

In this paper, we describe the LEAP submission to SRE19 challenge. All the submitted systems were based on the x-vector approach derived from extended TDNN (E-TDNN) models. We use three different E-TDNN models which were trained with various subsets of the training data. The x-vector extraction was followed by a back-end modeling. The vanilla system developed for

SRE19 used the conventional approach to back-end modeling for the x-vectors. This consisted of various normalization and dimensionality reduction techniques like the Within Class Covariance Normalization [15], length normalization [16], Linear Discriminant Analysis (LDA), Local pairwise LDA [17]. These normalized and dimensionality reduced x-vectors were modelled with PLDA for computing log-likelihood ratios.

For the back-end modeling, we explored a full neural network approach where the LDA, WCCN and length-normalization were incorporated as shared layers in a Siamese neural network [18]. This neural network operates on a pair of x-vectors and outputs a verification score. The NPLDA architecture was designed to perform the equivalent operations involved in the conventional PLDA back-end. A linear layer is used to replicate centering and LDA transformation, a length-norm layer for unit length normalization, and a quadratic layer which replicates the PLDA score (LLR) formula. However, the advantage of the proposed neural back-end is the ability to train the model using a discriminative cost function. The objective function was constructed to minimize the normalized minimum detection cost function C_{min}, similar to the efforts in [19]. Finally, following the score generation, we performed calibration and fusion of various systems using the techniques similar those in our previous SRE18 submission [2].

The rest of the paper is organized as follows: The SRE19 dataset details, the cost metric and training datasets used in our submission are detailed in Section 2. In Section 3, we give an overview of the x-vector feature extraction in three different training configurations using sub-sets of the training data. The back-end approaches using a traditional approach as well as the proposed neural network approach are detailed in Section 4. The experiments on the individual systems and calibration and fusion developed for SRE19 are reported in Section 5. This is followed by Section 6 where we report the analysis and improvements in the post-eval experiments. Finally, a summary of the paper is provided in Section 7.

2. SRE19 : Datasets and cost metric

Given a segment of speech and the target speaker enrolment data, the speaker verification task is to automatically determine whether the target speaker is present in the test segment. A test segment along with the enrolment speech segment(s) constitutes a *trial*. The system is required to process each trial independently and to output a log-likelihood ratio (LLR) score for that trial. The LLR for a given trial including a test segment u is defined as follows:

$$LLR(u) = \log\left(\frac{P(u|H_0)}{P(u|H_1)}\right) \tag{1}$$

where $P()$ denotes the probability density function (pdf), and H_0 and H_1 represent the null (i.e., u is spoken by the enrolled speaker) and alternative (i.e., u is not spoken by the enrolled speaker) hypotheses, respectively.

The test segment can range from 10 to 60 seconds, and all the trials are gender matched. For a given application, a decision is made by applying a certain application specific threshold to the log-likelihood ratio.

2.1. Performance metrics

The normalized detection cost function (DCF) is defined as

$$C_{Norm}(\beta, \theta) = P_{Miss}(\theta) + \beta P_{FA}(\theta) \tag{2}$$

where P_{Miss} and P_{FA} are the probability of miss and false alarms computed by applying detection threshold of θ to the log-likelihood ratios. The primary cost metric of the NIST SRE18 for the Conversational Telephone Speech (CTS) is given by

$$C_{primary} = \frac{1}{2}\left[C_{Norm}(\beta_1, \log\beta_1) + C_{Norm}(\beta_2, \log\beta_2)\right] \tag{3}$$

where $\beta_1 = 99$ and $\beta_2 = 199$.

The minimum detection cost (minDCF or C_{min}) is computed by using the detection thresholds that minimize the detection cost.

$$C_{min} = \min_{\theta_1,\theta_2} \frac{1}{2}\left[C_{Norm}(\beta_1, \theta_1) + C_{Norm}(\beta_2, \theta_2)\right] \tag{4}$$

The Equal Error Rate (EER) is the value of P_{FA} or P_{Miss} computed at the threshold where $P_{FA} = P_{Miss}$. We report the results in terms of EER, C_{Min} and $C_{primary}$ for all our systems.

2.2. Dataset

The training and development datasets used in our systems is summarized in Table 1. This choice of datasets is based on fixed condition training requirements mentioned in the evaluation plan [1]. The Voxceleb dataset and the previous SRE evaluation datasets along with the Switchboard corpus represent the major components of the training data. We had three variants of x-vector model training. While the Voxceleb dataset is primarily used for x-vector training, the other datasets were used for back-end (PLDA) training. The total duration of these datasets is about 11k hours of speech and it includes about 15k speakers. The SRE18 development set is held out from rest of the training for model fusion and hyper-parameter selection. The score normalization and calibration used the SRE18 unlabeled dataset. The SRE19 evaluation consisted of $2,688,376$ trials from $14,561$ segments. The mean duration of the recordings was around 60 seconds. More details about the SRE19 evaluation data can be found in the evaluation plan [1].

Table 1: Details of the training and development datasets used in the SRE19 evaluation. We indicate the data partitions used in the three x-vector systems (XV1-XV3) and the back-end models of the seven individual systems submitted (A-G).

Dataset	# Speakers	# Utterances	#Hours	X-Vector Training			Backend Training						
				XV1	XV2	XV3	A	B	C	D	E	F	G
Voxceleb 1 & 2	7323	1276888	2781	✓		✓							
Mixer 6	591	187197	2068		✓	✓		✓				✓	
SRE 04-06	2238	13346	1114		✓	✓		✓		✓		✓	
SRE 08	1336	9640	684		✓	✓	✓	✓		✓		✓	✓
SRE 10	446	15561	1272		✓	✓		✓		✓		✓	
Switchboard Corpus	2594	28181	2457		✓	✓		✓		✓		✓	
SRE 16 evaluation	201	10496	256		✓	✓	✓	✓			✓	✓	
SRE18 evaluation	188	13451	258		✓	✓		✓	✓		✓	✓	✓
SRE18 dev labelled	25	1741	34		✓	✓	Fusion, Calibration						
SRE18 dev unlabelled	-	2332	72				Score Normalization (as-norm)						

3. Front-end modeling

We trained three x-vector models with different subsets of the training data using the extended time-delay neural network architecture described in [20]. For x-vector extraction, an extended TDNN with 12 hidden layers and rectified linear unit (RELU) non-linearities is trained to discriminate among the speakers in the training set. The first 10 hidden layers operate at frame-level, while the last 2 operate at segment-level. There is a 1500-dimensional statistics pooling layer with between the frame-level and segment-level layers that accumulates all frame-level outputs from the 10th layer and computes the mean and standard deviation overall frames for an input segment. After training, embeddings are extracted from the 512-dimensional affine component of the 11th layer (i.e., the first segment-level layer). More details regarding the DNN architecture (e.g., the number of hidden units per layer) and the training process can be found in [20].

3.1. VoxCeleb x-vector system (XV1)

3.1.1. Training Datasets

The x-vector extractor is trained entirely using speech data extracted from combined VoxCeleb 1 and 2 corpora [21]. These datasets contain speech extracted from celebrity interview videos available on YouTube, spanning a wide range of different ethnicity, accents, professions, and ages. For training the x-vector extractor, we use about 1.2M segments from 7323 speakers selected from VoxCeleb 1 (dev and test), and VoxCeleb 2 (dev).

3.1.2. Feature Configuration and Model Description

The XV1 x-vector extractor was trained using 23 dimensional Mel-Frequency Cepstral Coefficients (MFCCs) from 25 ms frames every 10 ms using a 23-channel mel-scale filter-bank spanning the frequency range 20 Hz - 3700 Hz. In order to increase the diversity of the acoustic conditions in the training set, a 5-fold augmentation strategy is used that adds four corrupted copies of the original recordings to the training list. The recordings are corrupted by either digitally adding noise (i.e., babble, general noise, music) or convolving with simulated and measured room impulse responses (RIR). The noise and RIR samples are freely available[1]. Augmenting the original data with the noisy versions gives 6.3M training segments for the combined VoxCeleb dataset.

3.2. SRE x-vector system (XV2)

3.2.1. Training Datasets

The x-vector extractor is trained using speech data extracted from SwitchBoard corpus, Mixer 6, SRE04-10, SRE16 evaluation set and SRE18 development and evaluation sets. We used with 0.5M recordings from 6217 speakers. The datasets were augmented with the 5-fold augmentation strategy similar to the previous model. The recordings are corrupted by either digitally adding noise (i.e., babble, general noise, music) or convolving with simulated and measured room impulse responses (RIR).

3.2.2. Feature Configuration and Model Description

This x-vector model used 30 dimensional MFCC features using a 30-channel mel-scale filterbank spanning the frequency range 200 Hz - 3500 Hz. All other hyperparameters were the same as the XV1 x-vector system. The E-TDNN x-vector system was trained using speakers that had more than 8 utterances per speaker.

[1] http://www.openslr.org

3.3. Full X-Vector System (XV3)

3.3.1. Training Datasets

By combining the Voxceleb 1&2 dataset with Switchboard, Mixer 6, SRE04-10, SRE16 evaluation set and SRE18 development and evaluation sets, we obtained with 2.2M recordings from 13539 speakers. The datasets were augmented with the 5-fold augmentation strategy similar to the previous models. In order to reduce the weighting given to the VoxCeleb speakers (out-of-domain compared to conversational telephone speech (CTS)), we also subsampled the VoxCeleb augmented portion to include only 1.2M utterances.

3.3.2. Feature Configuration and Model Description

This x-vector model uses 30 dimensional MFCCs using a 30-channel mel-scale filterbank spanning the frequency range 20 Hz - 3700 Hz. All other hyperparameters were kept the same as the first x-vector system.

4. Back-end modeling

4.1. Generative PLDA (GPLDA)

The primary baseline we use to benchmark our systems is the Probabilistic Linear Discriminant Analysis (PLDA) [22] back-end implementation in the Kaldi toolkit [23]. This PLDA model is based on the two-covariance modeling approach. In order to train model, the x-vectors are centered, dimensionality reduced using Linear Discriminant Analysis (LDA), followed by unit length normalization [16]. These processed x-vectors are then used to train the PLDA model.

During the training, the GPLDA implementation computes a linear transform to center and simultaneously diagonalize the within and between class covariance of the training embeddings. These pre-processing steps are summarized as follows:

$$\underset{\text{x-vector}}{\boldsymbol{x}_r} \xrightarrow[\text{LDA}]{\text{Centering,}} \boldsymbol{y}_r \xrightarrow[\text{Normalization}]{\text{Unit Length}} \hat{\boldsymbol{y}}_r \xrightarrow{\text{Diagonalizing Transform}} \underset{\text{pre-processed embedding}}{\boldsymbol{\eta}_r}$$

The PLDA model on the processed x-vector for a given recording is,

$$\boldsymbol{\eta}_r = \Phi\boldsymbol{\omega} + \boldsymbol{\epsilon}_r \tag{5}$$

where $\boldsymbol{\eta}_r$ is the x-vector for the given recording, $\boldsymbol{\omega}$ is the latent speaker factor with a prior of $\mathcal{N}(0, I)$, Φ characterizes the speaker sub-space matrix. The across class covariance matrix (which captures across speaker variability) is denoted by $\Sigma_{ac} = \Phi\Phi^\intercal$. $\boldsymbol{\epsilon}_r$ is the residual term with distribution $\mathcal{N}(0, \Sigma_{wc})$ which is intended to capture session variability such as language, channel, noise, etc.

We denote the pre-processed embeddings of the enrolment and test segments as $\boldsymbol{\eta}_e$ and $\boldsymbol{\eta}_t$ respectively. The PLDA log-likelihood ratio is computed as

$$s(\boldsymbol{\eta}_e, \boldsymbol{\eta}_t) = \boldsymbol{\eta}_e^\intercal Q \boldsymbol{\eta}_e + \boldsymbol{\eta}_t^\intercal Q \boldsymbol{\eta}_t + \boldsymbol{\eta}_e^\intercal P \boldsymbol{\eta}_t + c \tag{6}$$

where,

$$Q = \Sigma_{tot}^{-1} - (\Sigma_{tot} - \Sigma_{ac}\Sigma_{tot}^{-1}\Sigma_{ac})^{-1} \tag{7}$$

$$P = \Sigma_{tot}^{-1}\Sigma_{ac}(\Sigma_{tot} - \Sigma_{ac}\Sigma_{tot}^{-1}\Sigma_{ac})^{-1} \tag{8}$$

with $\Sigma_{tot} = \Sigma_{ac} + \Sigma_{wc}$. Here, c is a constant term independent of the trial arising from the parameters of the latent variable distributions.

4.2. Neural PLDA

In the proposed pairwise discriminative PLDA model (neural PLDA), we pose the pre-processing steps and the log-likelihood ratio computation steps in the generative modeling as a function learnable in a neural network framework (Fig. 1). Specifically, we implement the pre-processing steps of centering and LDA as an affine layer. The unit-length normalization is implemented as a non-linear activation and PLDA centering and diagonalizing transform is implemented as another affine layer. Finally, the PLDA log-likelihood ratio given in Eq. 6 is implemented as a quadratic layer as shown in Fig. 1. Thus, the neural PLDA (NPLDA) implements the pre-processing of the x-vectors and the PLDA scoring as a neural back-end. The model parameters of the NPLDA are initialized with the baseline system and these parameters are learnt in a backpropagation setting.

4.2.1. Loss Function

The probability of miss and false alarms in Eq. 2 computed by applying a detection threshold θ are,

$$P_{Miss}(\theta) = \frac{\sum_{i=1}^{N} t_i \mathbb{1}(s_i < \theta)}{\sum_{i=1}^{N} t_i} \tag{9}$$

$$P_{FA}(\theta) = \frac{\sum_{i=1}^{N} (1 - t_i) \mathbb{1}(s_i \geq \theta)}{\sum_{i=1}^{N} (1 - t_i)} \tag{10}$$

Here, s_i is the score output by the model, t_i is the ground truth variable for trial i. That is, $t_i = 0$ if trial i is a target trial, and $t_i = 1$ if it is a non-target trial. $\mathbb{1}$ is the indicator function. The normalized detection cost function (Eq. 2) is not a smooth function of the parameters due to the step discontinuity induced by the indicator function $\mathbb{1}$. We propose a differentiable approximation of the normalized detection cost by approximating the indicator function with a warped sigmoid function similar to the efforts in [19] applied for text dependent end-to-end speaker verification.

$$P_{Miss}^{(\text{soft})}(\theta) = \frac{\sum_{i=1}^{N} t_i \left[1 - \sigma(\alpha(s_i - \theta))\right]}{\sum_{i=1}^{N} t_i} \tag{11}$$

$$P_{FA}^{(\text{soft})}(\theta) = \frac{\sum_{i=1}^{N} (1 - t_i) \sigma(\alpha(s_i - \theta))}{\sum_{i=1}^{N} (1 - t_i)} \tag{12}$$

Table 2: Impact of adding in-domain and out-of-domain data in the training of backend models. The front-end x-vectors for these models come from the XV1 system.

Model	Train Datasets	SRE18 Dev		SRE18 Eval		SRE19 Eval	
		EER (%)	C_{Min}	EER (%)	C_{Min}	EER (%)	C_{Min}
GPLDA	SRE 04-10, SWBD, MX6	11.4	0.65	13.2	0.70	13.3	0.69
GPLDA	+ SRE16	10.0	0.58	11.4	0.65	12.2	0.66
GPLDA	+ SRE18 Eval	**8.73**	0.56	6.93	0.51	9.63	0.58
NPLDA	SRE 04-10, SWBD, MX6	10.8	0.60	10.1	0.64	10.7	0.64
NPLDA	+ SRE16	10.9	0.53	9.61	0.61	10.4	0.63
NPLDA	+ SRE18 Eval	9.53	**0.49**	**6.73**	**0.48**	**8.64**	**0.55**

By choosing a large enough value for warping factor α, the approximation can be made arbitrarily close to the actual detection cost function for a wide range of thresholds.

We approximate P_{Miss} and P_{FA} terms in the primary cost metric (Eqn. 3) of the NIST SRE18 (CTS) with their soft counterparts to obtain a differentiable loss function

$$\mathcal{L}_{Primary} = \frac{1}{2}\left[C^{(\text{soft})}_{Norm}(\beta_1, \theta_1) + C^{(\text{soft})}_{Norm}(\beta_2, \theta_2)\right] \tag{13}$$

We train the pairwise NPLDA model with this differentiable cost function computed over gender matched trials. The proposed loss function in Eq.(13) is novel compared to previous attempts at discriminative modeling for speaker recognition using triplet loss or binary cross entropy loss.

4.2.2. *Sampling of Trials and NPLDA Training*

The procedure to sample trials is similar to what we used for the pairwise Gaussian Back-end model in our previous work [2]. We randomly sample pairs of gender matched x-vectors from each dataset belonging to target and non-target trials. Along with these manually sampled trials, we also include the SRE08,SRE10 and SRE16 evaluation trials from conversational telephone speech condition. This generates a total of 5M trials for the NPLDA training.

Unlike the cross entropy loss which is the negative log-likelihood of the labels, the soft DCF requires estimating $P^{(\text{soft})}_{Miss}$ and $P^{(\text{soft})}_{FA}$ for each batch. As any imbalance in target to non-target trial ratio in the mini-batches impacts the NPLDA model training, we choose a large batch size for training the NPLDA network. The implementation of the NPLDA can be found here[2].

4.3. Comparing Backend Models

Using the same front-end embedding extractor (XV1 configuration), we compare the two backend approaches based on generative GPLDA model as well as the neural PLDA model. These results are reported in Table 2. We train GPLDA models with Kaldi using SRE 04-10, switchboard corpus and Mixer6. To this, we add SRE16 and SRE18 data to study the improvements of adding more data. We then initialize the NPLDA model using the GPLDA back-end parameters and retrain the model using the cost function proposed. Table 2 summarizes the performance of these systems. In all the cases, the NPLDA yields significant improvements over PLDA. The improvements are more significant for the SRE cost function (C_{Min}) as the model is optimized for that metric.

[2] https://github.com/iiscleap/NeuralPlda

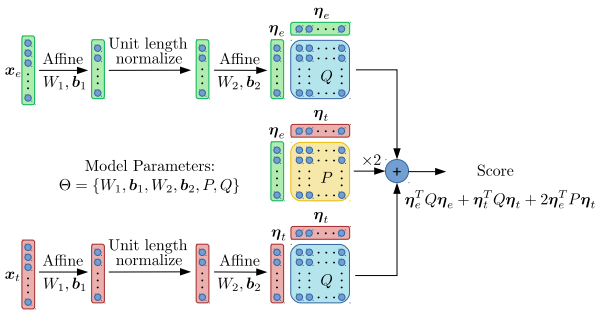

Figure 1: NPLDA model architecture: The two inputs $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are the enrollment and test x-vectors which constitute a *trial*.

5. Systems Submitted

As mentioned in the previous section, we had three different x-vector extraction models and two different back-end modeling approaches. In addition, several subsets of the training data were optionally used in the backend training. The overview of these systems is given in Table 1.

- System A: We use the XV1 x-vectors. The GPLDA model is trained using SRE16 eval set and the SRE08 dataset.
- System B: We use the x-vectors from full x-vector (XV3) model and train a NPLDA model using the Switchboard, Mixer 6 and SRE datasets, including

Table 3: Performance of the individual systems developed for SRE19 evaluation and the fusion system. The individual system results were obtained with the help of the keys for SRE19 provided by NIST after the evaluations. The best individual system is also highlighted. The description of systems A-G can be found in Table 1.

System	Front-end	Backend	Backend Train Datasets	SRE18 Dev		SRE19 Eval	
				EER (%)	C_{Min}	EER (%)	C_{Min}
A	XV1	GPLDA	SRE16, SRE08	10.6	0.6	11.1	0.66
B	XV3	NPLDA	SWBD, SRE(04-16), MX6, SRE18 Eval	**5.31**	**0.32**	**4.97**	**0.42**
C	XV1	GPLDA	SRE18 Eval	7.61	0.48	7.36	0.55
D	XV2	GPLDA	SWBD, SRE(04-10)	10.2	0.56	11.7	0.66
E	XV3	GPLDA	SRE16, SRE18 Eval	6.07	0.38	5.81	0.45
F	XV3	GPLDA	SWBD, SRE(04-16), MX6, SRE18 Eval	7.1	0.44	7.04	0.50
G	XV3	GPLDA	SRE08, SRE18 Eval	6.87	0.39	5.65	0.43
B+C	-	-	-	-	-	4.43	0.38
B+G	-	-	-	-	-	4.18	0.36

SRE 04-10, 16 and 18. We apply a sigmoid non-linearity at the output and optimize the proposed soft detection loss function.

- System C: We use the XV1 x-vectors with a GPLDA model trained with SRE18 evaluation set.
- System D: The XV2 x-vectors were used along with the GPLDA model trained using the Switchboard and SRE datasets.
- System E: We use the x-vectors from full XV3 system and the GPLDA model is trained using SRE16 and SRE18 evaluation dataset.
- System F: We use the x-vectors from the XV3 model and the GPLDA model is trained using the Switchboard, Mixer 6 and SRE datasets, including SRE 04-10, 16 and SRE18 evaluation dataset.
- System G: We use the x-vectors from the XV3 model and the GPLDA model is trained using the SRE08 dataset, SRE18 evaluation dataset only.

5.1. Calibration and Fusion

A linear score fusion of the different systems is done using the FoCAL toolkit [24], where the weights and biases are obtained with a logistic regression objective using a held-out set (SRE18 development set). In our experiments, we performed fusion of System B (NPLDA) and C, and the fusion of System B and G using the above mentioned approach. The systems for fusion were selected based on the complementary nature of training methods and datasets. The results on SRE19 evaluation set using the fused system scores are listed in Table 3.

For SRE19 submission, we attempted to calibrate the scores of the final systems using an affine transform which normalizes the within class score variance. The scores were then mean shifted such that the threshold corresponding to the minimum cost was moved to the target operating point (the operating point for actual cost is given in NIST SRE18 evaluation plan [1]). This was performed so as to minimize the difference between C_{min} and $C_{primary}$ on the SRE18 development dataset. This resulted in the $C_{primary}$ for our submission systems to be far from the minimum cost C_{min}, and hence we have not reported this in Table 3. In Section 6, we analyze the issues with this approach for calibration and highlight the steps we have taken to improve the calibration.

5.2. Summary of Results

The results obtained for the individual systems is given in Table 3. The best individual system was the combination of the XV3 x-vector extractor with the proposed NPLDA model. The full x-vector system (XV3) performs significantly better than the VoxCeleb (XV1) and the SRE (XV2) systems for any choice of back-end. The SRE 18 evaluation set is the closest to the SRE18 Dev and SRE19 Evaluation data (Tunisian Language). Comparing systems F and G implies that as we add more out of domain data like the older SRE data, switchboard and Mixer 6 in addition to the in domain (SRE18 Eval) data for PLDA training, the performance starts to degrade. Systems B (NPLDA) is trained with the same data as System F, and it is observed that it models both in-domain and out-of-domain data better than the GPLDA.

6. Post-eval Experimenents and Analysis

6.1. Calibration

In our previous work for SRE18 [2], we proposed an alternative approach to calibration, where the target and non-target scores are modelled as Gaussian distribution with a shared variance. The calibration procedure in this

Table 4: Performance of post-eval systems using improved calibration and adaptive score normalization (AS-Norm).

Model	Train Datasets	SRE18 Dev		SRE19 Eval		
		EER (%)	C_{Min}	EER (%)	C_{Min}	$C_{primary}$
GPLDA (XV1)	SRE08, SRE18 Eval	7.38	0.50	7.61	0.54	0.59
	+ AS-NORM	6.66	0.38	6.73	0.45	0.49
GPLDA (XV3)	SRE08, SRE18 Eval	6.87	0.39	5.65	0.43	0.56
	+ ASNORM	4.86	0.31	4.83	0.37	0.42
NPLDA (XV3)	SWBD, SRE(04-16), SRE18 Eval	4.88	0.28	4.56	0.39	0.47
	+ AS-NORM	**4.73**	**0.27**	**4.51**	**0.36**	**0.39**
Fusion B+G	+ AS-NORM	-	-	4.22	0.34	0.39

case involved the shifting of scores so that the threshold corresponding to the minimum cost on the development set is the point where the actual cost is computed on the evaluation trials ($\log \beta$ of [1]). This was done with the assumption that the score distributions of the development and evaluation trials match closely. Thus, the threshold where C_{min} is achieved in the development set may potentially match with evaluation trials. In the case of SRE18 evaluation [2], the development and evaluation score distributions were more or less the same, and the threshold that minimized the detection cost were very close. However, in SRE19, there was no exclusive matched development dataset provided. Hence, aforementioned calibration method using the SRE18 development dataset apploed on the SRE19 evaluation trials (as done for our submitted systems) turned out to be ineffective. Given the keys for SRE19 evaluation, we performed a score analysis and this is shown in Figure 2. As seen here, the computation of $C_{primary}$ using the distribution of SRE18 resulted in a sub-optimal calibration of the scores. In the post-eval efforts, we have performed score calibration based on the approach described in [25]. As seen in the plot, this matched the primary cost metric $C_{primary}$ (actDCF) closely with the minimum cost. The results for other individual systems based on the updated score calibration are reported in Table 4.

6.2. Score Normalization

We experiment with various cohort based normalization techniques [26, 27] using the SRE18 dev unlabelled set as the cohort. The best improvements were observed with the adaptive symmetric normalization (referred to as AS-Norm Type 1 in [27]). We achieve 24% relative improvement for the VoxCeleb x-vector system (XV1) and 21% relative improvement for the full x-vector system (XV3) on SRE18 development set. We achieve a comparatively lower but consistent improvements of about 15% on an average for the SRE19 evaluation set for all the systems. The improved results are summarized in Table 4.

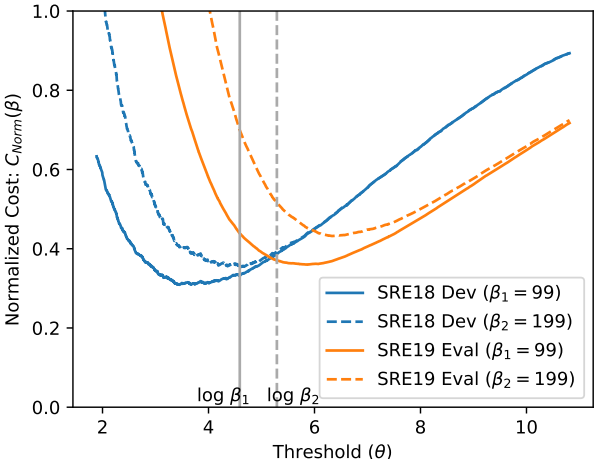

Figure 2: Plot illustrating the mismatch in the score distributions in SRE18 dev set and the SRE19 eval set.

7. Summary and Conclusions

In this paper, we provide an account of our efforts for the NIST SRE19 CTS challenge. We train three x-vector extractors and back-end models on different partitions of the available datasets, and report the performance of the individual as well as the fusion systems.

We explore a novel discriminative back-end model (NPLDA) inspired from deep neural network architectures and the generative PLDA model. For this model, we optimize a differentiable loss function constructed to approximate the detection cost function. Using a single elegant architecture targeted to optimize the speaker verification loss, the NPLDA uses a pair of x-vectors to directly generate the score. We provide analysis to show that NPLDA significantly boosts the performance of the system over the GPLDA for various datasets.

We discuss the errors that can be caused by calibration with a mismatched development set, and report the gains that can be achieved by using a cohort based adaptive score normalization technique for various systems.

8. References

[1] Omid Sadjadi, "NIST 2019 Speaker Recognition Evaluation: CTS Challenge - Evaluation Plan," .

[2] Shreyas Ramoji, Anand Mohan, Bhargavram Mysore, Anmol Bhatia, Prachi Singh, Harsha Vardhan, and Sriram Ganapathy, "The LEAP Speaker Recognition System for NIST SRE 2018 Challenge," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5771–5775.

[3] Douglas A Reynolds, Thomas F Quatieri, and Robert B Dunn, "Speaker Verification using Adapted Gaussian Mixture Models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[4] Patrick Kenny, Gilles Boulianne, Pierre Ouellet, and Pierre Dumouchel, "Joint Factor Analysis Versus Eigenchannels in Speaker Recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[5] Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-End Factor Analysis For Speaker Verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[6] Yun Lei, Nicolas Scheffer, Luciana Ferrer, and Mitchell McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1695–1699.

[7] Seyed Omid Sadjadi, Sriram Ganapathy, and Jason W Pelecanos, "The IBM 2016 Speaker Recognition System," in *Odyssey 2016*, 2016, pp. 174–180.

[8] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-vectors: Robust DNN Embeddings for Speaker Recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[9] William M Campbell, Douglas E Sturim, and Douglas A Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE signal processing letters*, vol. 13, no. 5, pp. 308–311, 2006.

[10] Sandro Cumani, Niko Brümmer, Lukáš Burget, Pietro Laface, Oldřich Plchot, and Vasileios Vasilakakis, "Pairwise Discriminative Speaker Verification in the i-Vector Space," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1217–1227, 2013.

[11] Sandro Cumani and Pietro Laface, "Generative pairwise models for speaker recognition," in *Odyssey*, 2014, pp. 273–279.

[12] Patrick Kenny, "Bayesian Speaker Verification with Heavy-Tailed Priors," in *Odyssey*, 2010, pp. 14–21.

[13] Ye Jiang, Kong Lee, Zhenmin Tang, Bin Ma, Anthony Larcher, and Haizhou Li, "PLDA Modeling in I-Vector and Supervector Space for Speaker Verification," in *Proc. Interspeech*, 2012, pp. 1680–1683.

[14] Sandro Cumani and Pietro Laface, "Large-Scale Training of Pairwise Support Vector Machines for Speaker Recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 22, no. 11, pp. 1590–1600, 2014.

[15] Andrew O Hatch, Sachin Kajarekar, and Andreas Stolcke, "Within-Class Covariance Normalization for SVM-based Speaker Recognition," in *Ninth international conference on spoken language processing*, 2006.

[16] Daniel Garcia-Romero and Carol Y Espy-Wilson, "Analysis of I-Vector Length Normalization in Speaker Recognition Systems," in *Proc. Interspeech*, 2011, pp. 249–252.

[17] Liang He, Xianhong Chen, Can Xu, Jia Liu, and Michael T Johnson, "Local Pairwise Linear Discriminant Analysis for Speaker Verification," *IEEE Signal Processing Letters*, 2018.

[18] Shreyas Ramoji, V Krishnan, Prachi Singh, Sriram Ganapathy, et al., "Pairwise Discriminative Neural PLDA for Speaker Verification," *arXiv preprint arXiv:2001.07034*, 2020.

[19] Victoria Mingote, Antonio Miguel, Dayana Ribas, Alfonso Ortega, and Eduardo Lleida, "Optimization of False Acceptance/Rejection Rates and Decision Threshold for End-to-End Text-Dependent Speaker Verification Systems," in *Proc. Interspeech 2019*, 2019, pp. 2903–2907.

[20] David Snyder, Daniel Garcia-Romero, Gregory Sell, Alan McCree, Daniel Povey, and Sanjeev Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5796–5800.

[21] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman, "VoxCeleb: A Large-Scale Speaker Identification Dataset," in *Proc. Interspeech*, 2017, pp. 2616–2620.

[22] Aleksandr Sizov, Kong Aik Lee, and Tomi Kinnunen, "Unifying Probabilistic Linear Discriminant Analysis Variants in Biometric Authentication," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2014, pp. 464–475.

[23] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely, "The kaldi speech recognition toolkit," 2011, IEEE Catalog No.: CFP11SRW-USB.

[24] Niko Brümmer, "FoCal Toolkit: MATLAB code for Evaluation, Fusion and Calibration of Statistical Pattern Recognizers," 2010.

[25] Niko Brümmer and Edward De Villiers, "The BOSARIS Toolkit: Theory, Algorithms and Code for Surviving the New DCF," *arXiv preprint arXiv:1304.2865*, 2013.

[26] Zahi N Karam, William M Campbell, and Najim Dehak, "Towards reduced false-alarms using cohorts," in *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2011, pp. 4512–4515.

[27] Pavel Matějka, Ondřej Novotný, Oldřich Plchot, Lukáš Burget, Mireia Diez Sánchez, and Jan Černocký, "Analysis of Score Normalization in Multilingual Speaker Recognition," in *Proc. Interspeech*, 2017, pp. 1567–1571.