I4U SYSTEM DESCRIPTION FOR NIST SRE'20 CTS CHALLENGE

Kong Aik Lee[1], *Tomi Kinnunen*[2], *Daniele Colibro*[3], *Claudio Vair*[3], *Andreas Nautsch*[4], *Hanwu Sun*[1], *Liang He*[5], *Tianyu Liang*[5], *Qiongqiong Wang*[6], *Mickael Rouvier*[7], *Pierre-Michel Bousquet*[7], *Rohan Kumar Das*[8], *Ignacio Viñals Bailo*[9], *Meng Liu*[10], *Héctor Deldago*[11], *Xuechen Liu*[2,12], *Md Sahidullah*[12], *Sandro Cumani*[3], *Boning Zhang*[5], *Koji Okabe*[6], *Hitoshi Yamamoto*[6], *Ruijie Tao*[8], *Haizhou Li*[8], *Alfonso Ortega Giménez*[9], *Longbiao Wang*[10], *Luis Buera*[11]

[1]Institute for Infocomm Research, A*STAR, Singapore
[2]School of Computing, University of Eastern Finland, Finland
[3]Loquendo and Politecnico di Torino, Italy
[4]EURECOM, France and vitas.ai, Germany
[5]Department of Electronic Engineering, Tsinghua University, China
[6]NEC Corporation, Japan
[7]LIA, Avignon University, France
[8]National University of Singapore, Singapore
[9]Vivolab, Spain
[10]Tianjian University, China
[11]Nuance Communications Inc., Spain
[12]INRIA, France

ABSTRACT

This manuscript describes the I4U submission to the 2020 NIST Speaker Recognition Evaluation (SRE'20) Conversational Telephone Speech (CTS) Challenge. The I4U's submission was resulted from active collaboration among researchers across eight research teams – I^{2}R (Singapore), UEF (Finland), VALPT (Italy, Spain), NEC (Japan), THUEE (China), LIA (France), NUS (Singapore), INRIA (France) and TJU (China). The submission was based on the fusion of top performing sub-systems and sub-fusion systems contributed by individual teams. Efforts have been spent on the use of common development and validation sets, submission schedule and milestone, minimizing inconsistency in trial list and score file format across sites.

1. INTRODUCTION

The I4U submission is a joint effort of cross-site research teams - I^{2}R (Singapore), UEF (Finland), Loquendo and Politecnico di Torino (Italy), VivoLab and Agnitio (Spain), NEC (Japan), THUEE (China), LIA (France), NUS (Singapore), INRIA (France) and TJU (China). The SRE'20 CTS Challenge is an extension to the SRE'19 CTS task with an additional challenge being multi-lingual evaluation set. We paid significant efforts in keeping a common training and development set, submission schedule and milestone, minimizing inconsistency in trial lists across sites. This manuscript presents the technical details of the datasets, sub-systems development, and fusion system.

We adopted a fusion strategy where sub-systems are fused with sub-fusion systems. These sub-systems are based on latest advances in neural speaker embeddings, namely, x-vector variants [1–6] and xi-vector [7]. All sub-systems developed in I4U adhere to the same development and validation sets as shown in Table 1. We created a common customized development set (I4U dev) comprising of sre18-dev and sre16-eval (30% trials) since there is no specified dev set in SRE'20. Most parts of the training set were provided by NIST and LDC as listed in Table 2. Given the open training condition in SRE'20, each site uses a slightly different training set, with additional data from various sources, which we describe in the following sections.

Table 1. Common data partitioning across all sites.

Partitions	Dataset
Development set	(a) sre18-dev, sre16-eval (30%)
Evaluation set	(b) sre20_cts_challenge
Train set	All datasets except (a) and (b)

Table 2. Train set used for open training condition.

Corpora	Source
sre16-eval (30%) SRE'18-dev, SRE'18-eval SRE'19-eval SRE'04, 05, 06, 08, 10, 12 Switchboard-2 Phase I & II & III Switchboard Cellular Part 1 & 2 Mixer6 Fisher 1 & 2	LDC, NIST
VoxCeleb 1 & 2 Private video data set	Open

2. FEATURE EXTRACTION

A summary of feature extractors of all subsystems is displayed in Table 3. We describe them in per-system fashion.

I2R/NUS/Vivolab: We consider mel frequency cepstral coefficient (MFCC) as feature for developing our system. For every utterance, 23-dimensional MFCC features are extracted considering short-term processing by using Hamming windowed frame of 25 ms with a shift of 10 ms. The feature extraction involves 23 mel filetbanks. In addition, the frames with sufficient voice activity are retained by performing an energy-based VAD.

LIA: For every utterances, 60-dimensional Filter-Banks (FBanks) features are extracted considering short-term processing by using Hamming windowed frame of 25 ms with a shift of 10 ms. A cepstral mean normalization is applied with a window size of 3 seconds. In order to removes silence and low energy speech segments, we used an energy-based VAD.

Loquendo and Politecnico di Torino: We extracted a single set of features for training all the Loquendo DNN embedding systems employed in this evaluation: 46 Log Mel Bands parameters with short time centering (STC) computed on both speech and non-speech audio frames. We used a VAD based on neural network (NN) phonetic decoding. The decoders are hybrid HMM-NN models trained to recognize 11 phone classes. The NN used for the VAD is a multi-layer perceptron that estimates the posterior probability of phonetic units, given an acoustic feature vector. It has been trained on several Speechdat corpora related to different European languages (e.g., English UK, French, German, Italian, Spanish, etc.) using approximately 600 hours of speech.

TJU: We consider log mel filterbanks (LogMEL) as feature for developing our system. For every utterance, 80-dimensional LogMEL features are extracted considering short-term processing by using Hamming windowed frame of 25 ms with a shift of 10 ms. Besides, sentence-level mean and variation normalization is emloyed for feature postprocessing.

UEF: 30-dimensional *Power-normalized cepstral coefficients* (PNCCs) are employed. Our feature extractor differs from the original work [8] by several means: 1) We use mel filterbanks [9] instead of Gammatone filterbanks, with number of triangular filters same as feature dimension. We do not observe notable difference on individual system performance in pilot experiments between the two; 2) We do not perform *discrete cosine transform* (DCT) as post-processing step on the output of power-law nonlinarity. Other operations and related parameters remain same as original work in [8].

3. EMBEDDINGS AND CLASSIFIERS

3.1. X-vector Neural embedding

3.1.1. Extended TDNN

The Extended-TDNN [2] neural network is an evolution of the classic x-vector network [1], increasing its depth along the frame-level processing block by alternating TDNN layers with feedforward ones. This network is trained according to two different losses: the traditional cross-entropy as well as *additive angular margin loss* [10].

3.1.2. Densely Connected TDNN

Densely connected time-delay neural network (D-TDNN) has been proposed in [3]. Its basic unit layer differs from the vanilla TDNN layer by cascading a fully-connected DNN and a TDNN with context expansion. Each of the layers are led by a ReLU activation. The network includes aggregated connection for multiple D-TDNN layers. Our implementation differs from the original network by two aspects: 1) We use attentive statistics pooling (ASP) instead of high-order statistics pooling; 2) When training the network we use additive angular margin softmax function, instead of cosine-based softmax.

3.1.3. ECAPA TDNN

The ECAPA TDNN architecture has been originally presented in [5]. We observed that the architecture is less accurate for telephonic speaker recognition than the other technological solutions considered, even if it is still moderately helpful for embeddings fusion. The main positive aspect is the fast training and inference time.

3.1.4. NEC 43-LAYERS RESNET

The NEC 43-LAYERS RESNET architecture has been proposed in [4]. This kind of DNN model is very deep, it includes many residuals connections, and it is accurate for telephonic speaker identification. We considered a few implementations with variations in the training and the DNN structure. The most notable solution exploits 53 instead of 43 layers.

Table 3. Summary of front-end processing components for each subsystem.

	I2R/NUS/Vivo	LIA	Loquendo	TJU	UEF
Feature set	MFCC	Fbanks	LogMEL	LogMEL	PNCC
# Features	23	60	46	80	30
# Filters	23	60	46	80	30
Framing	25ms/10ms	25ms/10ms	25ms/10ms	25ms/10ms	25ms/10ms
Window	Hamming	Hamming	Hamming	Hamming	Hamming
SAD	Energy	Energy	DNN	Energy	Energy
NFFT	512	512	256	400	512
Postprocessing	x	Mean	Short-Time Centering	Mean & Variation Norm	x

3.1.5. JHU MAGNETO 34-LAYERS RESNET

The JHU MAGNETO 34-LAYERS RESNET topology is described in [11]. It is based on modified 2D-CNN ResNet-34 architecture. It is by far the slower solution both for training and run-time inferencing because of the presence of 2D Convolutions, whereas all the other considered topologies use 1D Convolution. However, this DNN can achieve extremely good accuracy, in particular for short-duration conditions. We trained many different variants for this architecture to fully exploit its potential. In general, we observed some variability in terms of accuracy, so that the weight initialization and the training hyperparameters seem to be important concerns.

3.1.6. FTDNN RESNET

The FTDNN RESNET is a deep ResNet architecture exploiting semi-orthogonal 'factorized' TDNN layers as proposed in [12]. This provides very good accuracy results for the SRE20 CTS challenge, and it is also much faster than the JHU MAGNETO 34-LAYERS RESNET, in terms of training and run-time computation.

3.2. Xi-vector Neural embedding

Xi-vector is the Bayesian counterpart of the x-vector, taking into account the uncertainty estimate. X-vectors, and similar forms of deep speaker embedding, do not consider the uncertainty of features. In a restricted sense, uncertainty is merely captured implicitly with empirical variance estimates at the utterance level. Consequently, they show low robustness against local and random perturbation which is the inherent property of speech utterances. On the technology front, xi-vector offer a simple and straightforward extension to x-vector. It consists of an auxiliary neural net predicting the frame-wise uncertainty of the input sequence. On the theoretical front, xi-vector integrates the Bayesian formulation of linear Gaussian model to speaker-embedding neural networks via the pooling layer. In one sense, xi-vector integrates the Bayesian formulation of the i-vector to that of the x-vector. Hence, we refer to the embedding as the xi-vector, which is pronounced as /zai/ vector. We refer interested readers to [7].

4. SUB-SYSTEM DESCRIPTION

Listed below are the descriptions of the component classifiers used for I4U fusion. Neural embeddings were used in all subsystems, with substantial enhancement in terms of network architecture and loss function.

4.1. I2R

4.1.1. Front-end

We used xi-vector embedding with a simple five-layer TDNN. The training set consists primarily of English speech corpora, which encompasses Switchboard, Fisher, and the MIXER corpora used in SREs 04 – 06, 08, and 10. We used 23-dimensional MFCCs with 10ms frameshift. Mean-normalization over a sliding window of 3s and energy-based VAD were then applied. Data augmentation [13] was performed on the training sets using the MUSAN dataset [14].

4.1.2. Back-end

Probabilistic linear discriminant analysis (PLDA) was used as the back-end. As in most state-of-the-art implementations, speaker embeddings were reduced to 200 dimensions via *linear discriminant analysis* (LDA) projection before PLDA.

4.2. LIA

4.2.1. Additional Training Corpora

The LIA Training corpora used for training the DNN models is composed of I4U corpora and Multilingual LibriSpeech (MLS) datasets [15].

4.2.2. Front-end

The x-vector extractor used in the SRE20 CTS challenge is based on wide version of the ResNet-34, that employs a modified version of Squeeze-and-Excitation (SE). The extractor was trained on LIA Training corpora, cut into 4-second chunks and augmented with noise, as described in [1] and available as a part of the Kaldi-recipe. The speaker embeddings are 256-dimensional and the loss is the angular additive margin with scale equal to 30 and margin equal to 0.2. The size of the feature maps are 256, 256, 512 and 512 for the 4 ResNet blocks. The SE layers are added to the first 2 ResNet blocks. We use stochastic gradient descent with momentum equal to 0.9, a weight decay equal to 2.10^-4 and initial learning rate equal to 0.2. The batch size was set to 128, however, training on 4 GPUs in parallel. The implementation is based on PyTorch and the model traning takes about 4 days.

4.2.3. Back-end

The back-end of the single system is based on cosine similarity measurement. We obtained on the CTS progress set : EER of 2.83%, minDCF of 0.129 and actDCF of 0.126.

4.3. Loquendo and Politecnico di Torino

4.3.1. Additional Training Corpora

Besides the common I4U corpora we also used the following datasets for training the DNN embedding models and the related classifiers:

- Appen Conversational Telephonic Corpora: 1828 speakers - languages: Bulgarian, Dutch, Hebrew, Croatian, Italian, Portuguese, Romanian, Russian, Turkish
- CN-Celeb: 352 speakers - language: Chinese Mandarin
- Fisher Spanish: 109 speakers – language: Spanish
- Multi-Language Conversational Telephone Speech 2011: 68 speakers - languages: American English, Arabic, Czech, Bengali, Dari, Hindi, Persian, Pashto, Slovak, Spanish, South-Asian English, Tamil, Turkish, Urdu
- Rusten:121 speakers - language: Russian

4.3.2. Front-end: DNN embedding systems

For the SRE20 CTS challenge, we leveragred large DNN embeddings with more than 20 million parameters to achieve the best possible accuracy. We used PyTorch and we trained eleven DNN embeddings related to the ECAPA TDNN, NEC 43-LAYERS RESNET, JHU MAGNETO 34-LAYERS RESNET, and FTDNN RESNET families, introducing variation in terms of the layers size, the training hyperparameters, and the AMSoftmax loss margin. Table 4 shows the four standalone DNN embeddings, whose scores have been provided to the I4U consortium for the final fusion.

Moreover, Table 5 includes the additional seven DNN embeddings systems that have been used in the embedding fusion systems, described in the next section.

Table 4. Main Loquendo DNN embedding systems

DNN model	DNN type	Size (millions)
LoqDNN6	JHU Magneto	21.1
LoqDNN9	FTDNN RESNET	23.3
LoqDNN12	FTDNN RESNET	54.7
LoqDNN18	JHU Magneto	28.4

Table 5. Additional DNNs used in embedding fusion systems

DNN model	DNN type	Size(millions)
LoqDNN1	JHU Magneto	19.3
LoqDNN2	ECAPA TDNN	23.3
LoqDNN3	NEC 43-RESNET	24.1
LoqDNN4	JHU Magneto	19.3
LoqDNN5	JHU Magneto	19.8
LoqDNN7	NEC 53-RESNET	32.5
LoqDNN8	FTDNN RESNET	23.3

4.3.3. Back-end

The embeddings related to the standalone Loquendo DNNs were transformed in sequence by whitening, LDA projection that reduces their dimensions to 200 and length normalization. The LDA for the Loquendo embedding fusion systems has been computed on all the speakers of the training set used by Loquendo.

The classifier that has been used for obtaining the scores is the *neural pairwise support vector machine* (NeuralPSVM). The classifier takes inspiration from a work carried out by people working in the LEAP lab of the Indian Institute of Science, Bengaluru [16], [17]. The main idea is to perform a neural refinement to pre-trained LDA, whitening, WCCN, and PSVM [18], [19] scoring matrices. Starting from the initial model, gradient descent iterations are performed by using a smoothed version of NIST DCF as an objective loss function, with two working points: P_{target} = 1% and P_{target} = 0.5%.

Based on the best practices for discriminative PSVM training, impostor pairs were selected among similar speakers. A similar-speaker score matrix was computed offline by using an available accurate text-independent speaker recognition model. The ratio of same-speaker / different-speaker pairs for training was set 16 / 240: this means that, for a given speaker, 16 pairs of utterances of the target speaker are randomly selected and included in the training list, along with 240 pairs of utterances of the selected speaker coupled with randomly selected utterances of the 240 most similar speakers. The batch size was set to a high value (40*4096–68*4096) to have enough impostor pairs for a reliable estimation of the cost function.

To improve the accuracy, the duration information is added to the embeddings used for the training. The NeuralPSVM classifier includes three additional duration factors related to the sum, the difference, and the minimum number of frames used for extracting the embeddings to compare. The weights of the duration factors are automatically learned by the neural refinement.

While the standard PSVM is trained on a subset (˜8000) of speakers across the available training corpora the NeuralPSVM classifier is trained on the full training sets including ˜14.7 thousand speakers and ˜1.8 million utterances, including the original and augmented segments. For the multi-site embedding fusion, we exploited the standard PSVM classifier trained on the subset of corpora used for the LDA.

All our systems but one do not include any score normalization. Only the Emb-fus-1 system exploits the "Cal-Norm" approach, which computes cohort-based statistics on the scores following the approach defined by the adaptive AS-Norm [20]. In this case, the mean and the impostor standard deviation values are weighted with predefined coefficients and subtracted from the raw scores as compensation factors.

Table 6 summarizes the LDA, embedding classifier, and score normalization used for the different systems.

Table 6. Scored systems information

System	LDA size	Classifier	Normalization
LoqDNN6	200	NeuralPSVM	none
LoqDNN9	200	NeuralPSVM	none
LoqDNN12	200	NeuralPSVM	none
LoqDNN18	200	NeuralPSVM	none
Emb-fus-1	400	NeuralPSVM	cal-norm
Emb-fus-2	450	NeuralPSVM	none
Emb-fus-3	450	PSVM	none

4.3.4. *Progress Set Results*

Table 7 summarizes the results of the seven systems provided to I4U consortium on the official SRE20 CTS progress set. The scores of the systems do not include any calibration step for transforming the scores into LLRs at this point.

Table 7. Performance on the SRE 2020 CTS Progress Set

System	EER %	MinDCF
LoqDNN6	3.18%	0.117
LoqDNN9	3.06%	0.099
LoqDNN12	2.94%	0.096
LoqDNN18	3.00%	0.108
Emb-fus-1	3.30%	0.092
Emb-fus-2	3.31%	0.109
Emb-fus-3	3.66%	0.112

4.3.5. *Computational And Memory Requirements*

We used proprietary software for processing the audio files, computing the VAD, and saving LogMel Bands STC information on feature files, excluding the non-speech audio signal portions. On an Intel Core i5-6600, 3.3 GHz server, the program that creates the feature files is ˜200 times faster than real-time and requires less than ˜15 MB of memory for computing the VAD and performing the feature extraction.

The feature files related to the audio segments to analyze are then processed with a python script invoking PyTorch for computing the DNN embeddings. We used a server including the Quadro P6000 NVIDIA GPU (having Intel(R) Xeon(R) Gold 625 as CPU) and the embedding extraction procedure exploits GPU computation. Table 8 includes the figures related to the *real-time factor* (RTF) between the processed speech duration and the actual computation time needed by the inference tool. Moreover, Table 8 reports also the GPU memory as reported by the API "nvidia_smi.nvmlDeviceGetMemoryInfo" for all the eleven DNNs prepared by our site for the NIST SRE20 CTS challenge.

Table 8. DNN Embeddings Memory and RTFs

DNN model	GPU memory	RTF
LoqDNN1	19.31 GB	625x
LoqDNN2	4.38 GB	2011x
LoqDNN3	5.25 GB	1318x
LoqDNN4	18.72 GB	700x
LoqDNN5	19.65 GB	647x
LoqDNN6	20.13 GB	608x
LoqDNN7	6.70 GB	990x
LoqDNN8	5.54 GB	1274x
LoqDNN9	5.54 GB	1271x
LoqDNN12	20.2 GB	565x
LoqDNN18	6.25 GB	789x

Finally, the scoring for this evaluation has been performed by using a non-optimized Python environment. The elapsed time for obtaining the raw PSVM scores for the ˜2.6 M trials is around a couple of minutes. Obtaining the Cal-Norm scores requires about less than 3 hours. It is worth noting, however, that this implementation has been used just for these experiments. It has been designed to favor flexibility over computational efficiency.

4.4. NEC

4.4.1. *Front-end*

We used a variant of x-vector extractors which had a 43 TDNN layers with residual connections. This is exactly same as the one shown in the work [4]. Here, 2-head attentive statistics pooling was used in the same way as in the paper.

Additive margin softmax loss was used for optimization. 512-dimension bottleneck features from the first segment-level layer was used as speaker embeddings.

4.4.2. Back-end

Heavy-tailed PLDA (HT-PLDA) was used as the back-end in our systems. NIST SRE 04-12 datasets were used for producing out-of-domain PLDA. On the other hand, we also trained in-domain PLDA using SRE16 and SRE18 set. X-vectors for in-domain PLDA training were centerized using SRE16, SRE18 and SRE19 set. Then we applied linear interpolation between the out-of-domain PLDA and the in-domain PLDA. Weights for both PLDAs were 0.5.

4.5. NUS

4.5.1. Front-end

We perform data augmentation on the training dataset, which comprise of previous editions of SRE datasets 2004-2016, Mixer6, Switchboard, Fisher and VoxCeleb1-2 datasets. The number of training utterances are doubled by adding noisy and reverberated versions of the clean utterances. For this purpose, music, speech and babble noise segments extracted from the MUSAN database is used [14]. The MFCC features of the augmented training set are extracted for training a TDNN model based on the architecture described in [1] to extract 512-dimensional embeddings.

4.5.2. Back-end

The back-end of the system considers a 150-dimensional LDA to reduce the dimension of the x-vectors followed by PLDA classifier to compute the likelihood scores. It is noted that the PLDA model is first adapted with SRE 2018 eval set before scoring, which we found to give a better result than that without domain adaptation. Additionally, score normalization is applied with adaptive s-norm technique (30% top scores) considering the SRE 2018 eval set.

4.6. THUEE

4.6.1. Front-end

Training data we used includes SRE04-10, MIXER6, Switchboard (SWBD), Voxceleb 1&2 and Fisher datasets. These datasets (i.e., SRE, SWBD, Voxceleb) are augmented by different folds for different systems after convolving with far-field Room Impulse Responses (RIRs), or by adding noise from the MUSAN corpus. MFCC as acoustic feature is extracted with a 25 ms window size and a time shift of 10 ms. The operations of data augmentation and feature extraction are dependent on Kaldi x-vector recipe. We used the extended factorized TDNN (EF-TDNN) as baseline model and did some extensions on it called EF-TDNN_LSTMP and EF-TDNN_SE respectively. EF-TDNN_LSTMP, compared with basic EF-TDNN, replace the 19th layer with *long short-term memory with recurrent project layer* (LSTMP) [21]. LSTMP is combination of two improved methods. One is to introduce a recurrent projection layer between the LSTM layer and the output layer. The other is to introduce another non-recurrent projection layer to increase the projection layer size without adding more recurrent connections. In our LSTMP layer, it has 1024-dimensional cell, 512 recurrent and non-recurrent projection dimensions. Besides, considering the channel attention mechanism from the squeeze-and-excitation (SE) block [22] and its previous performance on different fields, EF-TDNN_SE mainly changes the structure of F-TDNN layer by adding SE layer with reduction ratio as 16 before the last convolutional layer.

4.6.2. Back-end

After the embeddings are extracted, they are then transformed to 300 dimension using LDA. Then, embeddings are projected into unit sphere. At last, adapted PLDA with no dimension reduction is applied. As for the data for LDA/PLDA adaptation, it is achieved by filtering more adaptive datasets according to the results on the leader board of CTS chanllenge.

4.7. TJU

4.7.1. Front-end

We use the SpeechBrain [23] toolkit to realize a standard ECAPA-TDNN system. After removing erroneous labels, 7,447 speakers are finally selected from SRE datasets 2004-2016 with 120,443 utterances. The network inputs 80-dimensional log fbank features and ouputs 192-dimensional speaker embeddings. Time domain SpecAugmentation, adding noise, adding reverberation, and adding noise & reverberation at the same time are implemented for data augmentation. Speeches are cropped and reunited after Voice Activity Detection (VAD). The whole process of features extraction are performed with GPUs using on-the-fly approach. Sentence-level normalization is used for input features. Additive Angular Margin Softmax and CyclicLRScheduler are employed to train the system.

4.7.2. Back-end

The back-end of the single system is based on cosine similarity measurement, with EER of 4.01%, MinDCF of 0.247, and ActDCF of 0.512 in the CTS eval set.

4.8. UEF

4.8.1. *Front-end*

Our speaker embedding extractor is based on the D-TDNN described in section 3.1.2. Our training data is composed of VoxCeleb1 training set, VoxCeleb2, LibriSpeech, Switchboard, MIXER-6, and SRE datasets (2004-2010). After removing very short utterances (less than 4 s) and speakers with too few number of utterances (less than 8), the total number of speakers are 12860. We perform data augmentation using *room impulse response* (RIR) [13] and MUSAN dataset [14], followed by random sampling a subset which is two times larger than the original data. We combined the subset and the original set as the training data for D-TDNN. Adam [24] is adopted as the optimizer. Speaker embeddings are extracted from the first fully-connected layer after pooling.

4.8.2. *Back-end*

The extracted speaker embeddings are mean subtracted and length normalized, before being transformed to 200 dimension using LDA. We used SRE2004-2010, Switchboard and MIXER-6 to train PLDA, and SRE20 development set for adapting it [25]. Log-likelihood scores provided by this subsystem are returned by the adapted PLDA.

4.9. Vivolab and Agnitio

4.9.1. *Front-end*

Vivolab embedding extractor is built based on the Extended-TDNN described in section 3.1.1. The training corpus consists of the traditional MIXER6 corpora (SRE04-06, SRE08 and SRE10), complemented with the data from SRE18 and VoxCeleb 1. Agnitio embedding extractor is also based on Extended-TDNN neural network trained with the same data as Loquendo and Politecnico di Torino do. As input features, Agnitio selected wav2vec [26].

4.9.2. *Back-end*

Vivolab backend is based on the state-of-the-art LDA-PLDA pipeline. For this purpose we define two subsets of data to take into consideration: On the one hand we include both MIXER6 and VoxCeleb1 data as a large out-of-domain subset. On the other hand the in-domain subsets consists of excerpts from SRE16, SRE18eval and SRE19. Due to the different nature between both subsets we first apply CORAL [27], weighting both domains with a factor of 0.5. The adapted embedings now undergo centering and LDA-based whitening (a reduction of dimension up to 300). Final scores take into account two different models, the simplified PLDA (SPLDA) as well as a joint PLDA of two factors [28], with a inter-speaker subspace dimension of 100. These models consider the same training corpus as CORAL and the LDA. Agnitio backend consists on LDA-PSVM trained with the same data as Loquendo and Politecnico di Torino do.

5. FUSION AND CALIBRATION

5.1. Embedding Fusion

To enhance the accuracy, we also exploited embedding fusion approaches by stacking x-vectors produced by different systems. This allows exploiting the orthogonality of different systems and obtaining better accuracy results. It is worth remarking that such an approach is computationally expensive, and it makes sense mainly on challenges and evaluations to showcase technology.

Currently, we provided scores to the I4U consortium related to three embedding fusions. Two of them were based on the fusion of Loquendo DNN embeddings systems. The remaining DNN embedding fusion involved using embeddings produced by multiple I4U sites. Table 9 summarizes the embedding fusion systems used in the challenge and their stacked size.

Similar to the embeddings related to the standalone Loquendo DNNs, fused embedding were transformed in sequence by whitening, LDA projection that reduces their dimensions (to 400 or 450) and length normalization. Moreover, the LDA for the Loquendo embedding fusion systems has been computed on all the speakers of the training set used by Loquendo, while for the multi-site embedding fusion the LDA has been computed on a common subset of corpora including NIST SRE04-10, Mixer6, SRE16, SRE18, and SRE19. Tables 6 and 7 compared the fused embeddings with that of standalone embeddings.

Table 9. Embedding fusion systems

Model Name	DNN models	Stacked size
Emb-fus-1	LoqDNN1-2-3-5	2048
Emb-fus-2	LoqDNN4-6-7-8-9	2304
Emb-fus-3	I2R,LOQDNN6-9, NEC,TJU,UEF	3072

5.2. Score Fusion and Calibration

For score fusion, a Python implementation[1] of the BOSARIS toolkit [29] was used for logistic regression yielding log-likelihood ratio (LLR) scores. In preliminary studies, we *(i)* investigated and extended quality estimates [30, 31], and *(ii)* observed that some systems are causing worse performance if included (especially on the progress set).

We found log-duration to be helpful despite data shifts. In Miranti's method [31], log-ratios of both audio segments' durations are investigated. We generalise and let the regression find exponents for each duration value. Of n systems, scores

[1]`https://gitlab.eurecom.fr/nautsch/pybosaris`

S_i are fused to S' using $n+3$ weights w_0, w_i, w_r, w_p and depending durations of reference (summed if multiple files) and probe audios d_r, d_p (also easier to implement):

$$S' = w_0 + \sum_{i \in 1..n} w_i\, S_i + w_r\, \log(d_r) + w_p\, \log(d_p). \quad (1)$$

When fusing all above described subsystems and embeddings fusion systems, we investigated the individual LLR contribution of each subsystem ($w_i\, S_i$). Looking at the absolute value of a system's highest/lowest contribution, we observed that some contributed small offsets of ≤ 1 to any LLR; systems of weak contribution. Jackknifing showed that some remained useful nonetheless. Our fusion comprised twelve subsystems and the two duration sets.[2]

Table 10. Initial analysis: LLR contribution potentials.

Subsystem	$\min w_i\, S_i$	$\max w_i\, S_i$	Used for fusion
Emd-fus-1	-6.3	2.6	x
Emb-fus-2	-9.0	5.7	x
Emb-fus-3	-6.8	2.6	x
I2R	-0.2	0.1	
I2Rr1	-2.6	1.1	
I2Rr2	-0.7	1.7	x
I2Rr3	-0.7	1.9	
I2Rr4	-8.1	3.5	x
LoqDNN6	-0.9	1.7	x
LoqDNN9	-0.1	0.1	
LoqDNN12	-3.9	2.2	x
LoqDNN18	-0.9	1.6	
LIA	-0.8	0.4	x
NEC	-4.8	2.5	x
THUEE	-2.6	1.3	x
UEF	-0.4	0.7	x
VIVO	-1.8	0.5	x

Table 10 shows the contribution to fusion of the above subsystems in a preliminary assessment fusion. Neither are durations included, nor is the decision for inclusion in the final fusion set-up made solely on this information. Yet, for this composition, one can identify the embedding fusions to be the draught horses here. While individual systems can perform good, their contribution here might be little when they are many alike systems in that particular composition (weights depend on the composition of systems to be fused); vice versa, systems of low individual performance might appear better here for they behave differently. Moreover, some systems of little contribution have critical impact on the small yet remaining performance gains on progress set.

[2]Since the challenge rules were that systems of lower actual DCF will become the new primary system, we added a -2 offset to all our scores, so we could update potentially later with lower minDCF systems.

The contribution to the I4U fusion is shown in Table 11. Weights are shown for curtosy reasons; some are negative—some systems became correctors to an otherwise overconfident fusion outcome (if they would not have been a part of the fusion). Log-durations are demonstrated for their need as a correction measure; as a quality estimate. Among the subsystems, the ranges of LLR contribution changed slightly. In comparison to the subsystems' contribution to the fusion LLR, log-duration appears not considerable to yield an LLR by itself; yet, it re-adjusts for too high LLR estimates. There are two impacts of weights: proportion a subsystem matters within the fusion composition while also making its scores fit to the LLR scale of composed systems.

Table 11. Fusion: LLR contributions ($w_0 = 12.5$).

Subsystem	$\min w_i\, S_i$	$\max w_i\, S_i$	weight $\times 100$
Emd-fus-1	-6.1	2.5	27.6
Emb-fus-2	-8.3	5.2	50.8
Emb-fus-3	-6.8	2.6	55.5
I2Rr2	-0.7	1.8	-3.5
I2Rr4	-7.2	3.1	13.5
LoqDNN6	-0.7	1.5	-13.1
LoqDNN12	-3.6	1.0	25.1
LIA	-0.9	0.4	0.5
NEC	-4.1	2.2	2.7
THUEE	-2.9	1.5	1.0
UEF	-0.5	0.9	-0.6
VIVO	-0.2	0.0	-0.8
$\log(d_r)$	-4.0	-3.3	-39.8
$\log(d_p)$	-4.7	-3.2	-52.7

6. RESULTS ON DEVELOPMENT AND PROGRESS SETS

Performance of the submissions on I4U Development set (as in Table 1) and Progress (sre20_progress) Sets are shown in Table 12.

Table 12. Performance of the primary fusion on I4U Development and SRE'20 Evaluation Sets.

Equalized	EER (%)	Min $C_{primary}$	Act $C_{primary}$
I4U Dev	2.18	0.038	0.043
Progress	2.53	0.077	0.094
Test	2.91	0.066	0.070

7. COMPUTATION AND MEMORY REQUIREMENT

The GPU and CPU time for individual sub-systems to process a single trial is shown in Table 13. Peak memory usage required to process a pair of enrollment and test recordings was approximately 300 to 700 MB.

Table 13. CPU and GPU execution time used to process a single trial for each sub-system/sub-fusion in term of real-time factor (RT).

Sub-system	RT (CPU)	RT (GPU)
Emd-fus-1	-	0.0044
Emd-fus-2	-	0.0056
Emd-fus-3	3.60	0.0094
I2R (r2, r4)	0.281	-
LoqDNN6	-	0.0015
LoqDNN12	-	0.0018
LIA	-	0.0017
NUS	-	0.0020
NEC	0.417	-
THUEE	-	-
UEF	2.902	0.0012
VIVO	-	-

8. CONCLUSION

The I4U submissions were based on the score fusion of multiple sub-systems. Scores from individual sub-system/sub-fusion were first calibrated followed by simple linear transformation fusion. Another highlight to I4U submission is embedding fusion, where sub-system embeddings are concatenated and used as input to the classifier.

9. ACKNOWLEDGEMENTS

This work was partially funded by the ANR/JST VoicePersonae project (grant No. JPMJCR18A6).

10. REFERENCES

[1] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2018*, 2018, pp. 5329–5333.

[2] David Snyder, Jesús Villalba, Nanxin Chen, Daniel Povey, Gregory Sell, Najim Dehak, and Sanjeev Khudanpur, "The jhu speaker recognition system for the voices 2019 challenge," in *Proc. Interspeech 2019*, 2019, pp. 2468–2472.

[3] Ya-Qi Yu and Wu-Jun Li, "Densely Connected Time Delay Neural Network for Speaker Verification," in *Proc. Interspeech 2020*, 2020, pp. 921–925.

[4] Kong Aik Lee, Koji Okabe, Hitoshi Yamamoto, Qiongqiong Wang, Ling Guo, Takafumi Koshinaka, Jiacen Zhang, Keisuke Ishikawa, and Koichi Shinoda2, "NEC-TT Speaker Verification System for SRE'19 CTS Challenge," in *Proc. Interspeech 2020*, 2020, pp. 2227–2231.

[5] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck, "ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification," in *Proc. Interspeech 2020*, 2020, pp. 3830–3834.

[6] Daniel Garcia-Romero, Greg Sell, and Alan Mccree, "MagNetO: X-vector Magnitude Estimation Network plus Offset for Improved Speaker Recognition," in *Proc. The Speaker and Language Recognition Workshop (Odyssey 2020)*, 2020, pp. 1–8.

[7] Kong Aik Lee, Qiongqiong Wang, and Takafumi Koshinaka, "Xi-vector embedding for speaker recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 1385–1389, 2021.

[8] C. Kim and R. Stern, "Power-normalized cepstral coefficients (PNCC) for robust speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 7, pp. 1315–1329, 2016.

[9] Xuechen Liu, Md. Sahidullah, and Tomi H. Kinnunen, "Optimized power normalized cepstral coefficients towards robust deep speaker verification," *ArXiv*, vol. abs/2109.12058, 2021.

[10] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," 2019.

[11] Daniel Garcia-Romero, Greg Sell, and Alan Mccree, "MagNetO: X-vector Magnitude Estimation Network plus Offset for Improved Speaker Recognition," in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, 2020, pp. 1–8.

[12] Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur, "Semi-Orthogonal Low-Rank Matrix Factorization for Deep Neural Networks," in *Proc. Interspeech 2018*, 2018, pp. 3743–3747.

[13] Tom Ko, Vijayaditya Peddinti, Michael Seltzer Daniel Povey, and Sanjeev Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *Proc. IEEE ICASSP*, 2017, pp. 5220–5224.

[14] David Snyder, Guoguo Chen, and Daniel Povey, “MUSAN: a music, speech, and noise corpus,” in *arXiv:1510.08484*, 2015.

[15] Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert, “Mls: A large-scale multilingual dataset for speech research,” *ArXiv*, vol. abs/2012.03411, 2020.

[16] Shreyas Ramoji, Prashant Krishnan V au2, Prachi Singh, and Sriram Ganapathy, “Pairwise Discriminative Neural PLDA for Speaker Verification,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2020*, 2020.

[17] Shreyas Ramoji, Prashant Krishnan, and Sriram Ganapathy, “Nplda: A deep neural plda model for speaker verification,” in *Odyssey 2020 The Speaker and Language Recognition Workshop*. Nov 2020, pp. 202–209, ISCA.

[18] Sandro Cumani and Pietro Laface, “Training pairwise support vector machines with large scale datasets,” in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1645–1649.

[19] Sandro Cumani and Pietro Laface, “Large-scale training of pairwise support vector machines for speaker recognition,” in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2014, vol. 22, pp. 1590–1600.

[20] Sandro Cumani, Pier Domenico Batzu, Daniele Colibro, Claudio Vair, Pietro Laface, and Vasileios Vasilakakis, “Comparison of speaker recognition approaches for real applications,” in *Proc. Interspeech 2011*, 2011, pp. 2365–2368.

[21] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition,” *ArXiv*, vol. abs/1402.1128, 2014.

[22] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and E. Wu, “Squeeze-and-excitation networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 2011–2023, 2020.

[23] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, Ju-Chieh Chou, Sung-Lin Yeh, Szu-Wei Fu, Chien-Feng Liao, Elena Rastorgueva, François Grondin, William Aris, Hwidong Na, Yan Gao, Renato De Mori, and Yoshua Bengio, “SpeechBrain: A general-purpose speech toolkit,” 2021, arXiv:2106.04624.

[24] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[25] P. ousquet and M. Rouvier, “On Robustness of Unsupervised Domain Adaptation for Speaker Recognition,” in *Proc. Interspeech 2019*, 2019, pp. 2958–2962.

[26] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli, “wav2vec: Unsupervised Pre-Training for Speech Recognition,” in *Proc. Interspeech 2019*, 2019, pp. 3465–3469.

[27] Baochen Sun, Jiashi Feng, and Kate Saenko, “Return of frustratingly easy domain adaptation,” in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. 2016, AAAI’16, p. 2058–2065, AAAI Press.

[28] Luciana Ferrer and Mitchell McLaren, “Joint plda for simultaneous modeling of two factors,” *Journal of Machine Learning Research*, vol. 20, no. 24, pp. 1–29, 2019.

[29] N. Brümmer and E. de Villiers, “The BOSARIS toolkit user guide: Theory, algorithms and code for binary classifier score processing,” Tech. Rep., AGNITIO Research, South Africa, 12 2011.

[30] Luciana Ferrer, Lukas Burget, Oldrich Plchot, and Nicolas Scheffer, “A unified approach for audio characterization and its application to speaker recognition,” in *Proc. Odyssey*, 2012.

[31] Miranti Indar Mandasari, Rahim Saeidi, and David van Leeuwen, “Calibration based on duration quality measures function in noise robust speaker recognition for NIST SRE’12,” in *Proc. Biometric Technologies in Forensic Science*, 2013.