IDIAP RESEARCH REPORT

IDIAP SUBMISSION TO NIST LRE22 LANGUAGE RECOGNITION EVALUATION

Amrutha Prasad Driss Khalil Srikanth Madikeri
Petr Motlicek

Idiap-RR-11-2025

OCTOBER 2025

Centre du Parc, Centre du Parc, Rue Marconi 19, CH - 1920 Martigny
T +41 27 721 77 11 F +41 27 721 77 12 info@idiap.ch www.idiap.ch

IDIAP SUBMISSION TO NIST LRE22 LANGUAGE RECOGNITION EVALUATION

Amrutha Prasad[†,‡] *Driss Khalil*[†] *Srikanth Madikeri* [†]
Petr Motlicek[†,‡]

[†] Idiap Research Institute, Martigny, Switzerland
[‡] Brno University of Technology, Brno, Czechia

ABSTRACT

The Idiap submission to the NIST Language Recognition Evaluation (LRE) 2022 consists of three types of systems: (i) Random Forest (RF) and Support Vector Machine (SVM) classifiers trained on embeddings obtained from a pre-trained model from SpeechBrain, (ii) Kaldi-based x-vector-PLDA (Probabilistic Linear Discriminant Analysis) system trained with Kaldi, and (iii) Kaldi-based PLDA trained on the previously mentioned pre-trained model's embeddings. The score-level fusion (that is, linear combination of scores) of the RF and SVM classifiers in (i) was submitted as the primary system for the fixed condition. The score-level fusion of (ii) and (iii) were used as the alternative system. For the open condition, we used two Kaldi-based x-vector PLDA systems with score-level fusion, where additional data from the BABEL corpora was used to train the PLDA models. Our models were developed with Kaldi, PyTorch, SpeechBrain, and Scikit-learn toolkits.

Index Terms— language identification, x-vector

1. INTRODUCTION

Given a speech segment and a target language, the goal of language recognition is to automatically determine if the target language was spoken in the segment. There are 14 target languages in the LRE '22 challenge. The output from the system is a set of score vectors – one 14-dimensional vector for each test segment[1]. The target languages with their language code are provided in Table 1.

We applied energy-based speech activity detection for the front-end systems and used Mel frequency cepstral coefficients (MFCCs) as input features. Our submission mainly consists of three back-end systems – two of which use the Kaldi-based [1] x-vector models [2], and the ECAPA-TDNN model [3] from SpeechBrain [4] trained with VoxLingua 107 data for language identification task as the third. In addition, we also investigated employing relatively simple classification systems such as Random Forest (RF) and Support Vector Machine (SVM) for language identification instead of the PLDA [5] classifier.

2. DATASETS

2.1. Training

2.1.1. Fixed condition

This section provides an overview of the LRE17, LRE22, and Voxlingua 107 datasets used in our experiments.

[1]Evaluation plan: `https://lre.nist.gov/uassets/3`

Table 1. The target languages of LRE '22 along with their language codes.

Target languages	Language Code
Afrikaans	afr-afr
Tunisian Arabic	ara-aeb
Algerian Arabic	ara-arq
Libyan Arabic	ara-ayl
South African English	eng-ens
Indian-accented South African English	eng-iaf
North African French	fra-ntf
Ndebele	nbl-nbl
Oromo	orm-orm
Tigrinya	tir-tir
Tsonga	tso-tso
Venda	ven-ven
Xhosa	xho-xho
Zulu	zul-zul

LRE17 Train set: the training subset of the LRE 2017 corpus, which comprises 14 languages[2] and has a total duration of 2066.5h.

LRE17 Development set: the development subset of the LRE 2017 corpus[3] which has a total duration of 62 h.

LRE22 Development set: this set has a duration of 30 h with 300 audio files per language. The data is split into train and development sets such that there is no overlap of audio-id in the subsets and has 150 audio files for each language.

The **VoxLingua107** dataset consists of short speech segments extracted from YouTube videos, and labeled according to the language of the video title and description, with some post-processing steps to filter out false positives [6]. It contains 6628 h of speech from 107 languages. As it is a relatively large corpus (compared to our other sets) with approximately 25 M utterances, we only used a subset of this data in our experiments, as shown in Table 2. We randomly selected 2000 audio files per language with a total duration of 545 h.

[2]LRE 2017 Evaluation plan: `https://www.nist.gov/system/files/documents/2017/09/29/lre17_eval_plan-2017-09-29_v1.pdf`

[3]LRE 2017 Evaluation plan: `https://www.nist.gov/system/files/documents/2017/09/29/lre17_eval_plan-2017-09-29_v1.pdf`

Table 2. Overview of the train and test splits of the datasets described in Section 2.

	Number of	Duration (h)		No. of segments (k)	
Dataset	**Languages**	Train	Test	Train	Test
LRE17 train	14	2061	-	15.3	-
LRE17 dev	14	30.4	30	1.8	1.8
LRE22 dev	14	15.3	14.5	2.1	2.1
Voxlingua 107	106	545.5	-	210	-
BABEL	13	703	-	634	-

Table 3. **List of languages used from IARPA BABEL dataset.**

Amharic	Assamese	Bengali	Cantonese
Cebuano	Dhuluo	Georgian	Guarani
Lao	Lithuanian	Mongolian	Pashto
Zulu			

2.1.2. Open condition

In addition to the data used in the fixed condition (as mentioed in Section 2.1.1), we also used a selected portion (due to data-processing time constraints) of the BABEL dataset. Table 3 provides an overview of the languages used. The total duration of the data used is 700h.

3. FIXED TRAINING

Two systems are submitted in this scenario.

3.1. Primary system

Our primary system is the score fusion of the two classifiers trained with the Scikit-learn [7] toolkit: RF and SVM. Embeddings generated with the SpeechBrain model are used as input to the classifiers.

Data: The train split of the LRE22 dev data is used for training, while its test split is used for tuning the classifiers.

Experimental setup: the publicly available[4] ECAPA-TDNN model from SpeechBrain is used to generate embeddings of 256-dimension. The model was trained using the VoxLingua107 for the language recognition task.

As we were working with a relatively small training set (2100 embeddings for training), we also explored classical machine learning algorithms that are more interpretable than neural networks. We used two approaches: one based on random forests and another based on Support Vector Machines (SVM) [8]. Random Forest is an ensemble learning method that builds multiple decision trees and helps avoid overfitting compared to a simple decision tree model [9]. The best model used the following parameters: estimators = 100, criterion = "gini", maximum depth = 11. The second approach based on the Support vector machine used the following parameters: RBF kerel with C = 10, gamma = 0.001.

GridSearch was used to generate the best parameters for both the models [10]. For our primary submission, the scores obtained from the RF and SVM systems were fused with a weight of 0.4 and 0.6, respectively, obtained once again by tuning on the development set.

[4] https://huggingface.co/speechbrain/lang-id-voxlingua107-ecapa

3.2. Alternate system

Our alternate submission was obtained from the score-level fusion of the two systems–with a weight of 0.5 for each–described in the next two subsections.

3.2.1. X-vector based PLDA

Data: LRE17 train data is used to train the x-vector system. The following data were used to train the PLDA: LRE17 train set, the train split of LRE17 dev set, and the train split of LRE22 dev data. The data are also additionally augmented by adding reverberation and noise using the Musan [11] corpus.

Experimental Setup: After down-sampling the speech data to 8 kHz, 20-dimensional MFCCs were extracted with a 25 ms window of speech data with a 10 ms frameshift. Band-pass filtering was applied between 20 to 3700 Hz. Log of energy was added to the feature vector, and these features were mean-normalized over a sliding window of 3 seconds. Energy-based voice activity detection (VAD) removes the non-speech frames. For training the x-vector system, a chunk-size between 80 to 120 speech frames is used [12]. The standard x-vector recipe from Kaldi [1] was used to train the extractor. The x-vector uses 7 Time Delay Neural Network (TDNN) layers with an input dimension of 20 and an output dimension of 512. The training is run for three epochs with an initial and final learning rate of 0.001 and 0.0001, respectively. After comparing different scoring techniques [13, 14] associated with a PLDA model, we used the score-averaging approach.

3.2.2. ECAPA-TDNN based PLDA

Data: the VoxLingua107 data subset was used for PLDA training, followed by the augmented version of the train split of LRE22 dev data for PLDA adaptation.

Experimental setup: the embeddings are generated from the SpeechBrain model for both the VoxLingua107 subset and the noise-augmented version of the training subset of the LRE22 dev data. We first trained a PLDA classifier on VoxLingua107 subset, and adapted to the target languages using Bayesian Maximum a Posteriori (MAP) estimation on the augmented version of the train split of the LRE22 dev set. The test split of the LRE22 dev set was used for initial evaluations, and selecting the scoring method. We used the x-vector averaging approach for the PLDA system [13, 14].

4. OPEN TRAINING CONDITION

4.1. Primary

Our primary submission for the open training condition was the score-level fusion of two x-vector based PLDA systems described next.

Data: LRE17 train data is used for training the x-vector system. Train splits of LRE22 dev data and BABEL datasets are then used to train two separate PLDA models. The train split of the LRE22 dataset is additionally augmented by adding reverberation and noise with the Musan [11] corpus. No data augmentation was applied to the BABEL dataset.

Experimental Setup: The x-vector uses 3 TDNN layers and 10 Factorized TDNN (TDNN-F) [15] with an input dimension of 20 and an output dimension of 512. The TDNN and TDNN-F layers have sizes of 1536 and a bottleneck dimension of 160. The training is run for four epochs with an initial and final learning rate of 0.00001 and 0.000015, respectively.

Table 4. Results on the test split of the LRE 22 development set of NIST LRE 2022 and evaluation set of for all systems presented as provided by the NIST toolkit. min C: minimum Decision Cost Function, act C: actual Decision Cost Function.

System	Test split of LRE22 dev set	
	min.C	act.C
Fixed Training		
Random Forest	0.56	0.56
Support Vector Machine	0.53	0.52
Fusion (**Primary**)	0.52	0.52
x-vector plda	0.70	0.72
speechbrain plda	0.62	0.82
Fusion (**Alternate**)	0.57	0.6
Open Training		
plda - LRE dev22	0.75	0.87
plda - BABEL	0.76	0.81
Fusion (**Primary**)	0.57	0.6
Support Vector Machine (**Alternate**)	0.53	0.52

We then trained the following 2 PLDA classifiers: (1) on the augmented version of the train split of LRE22 dev with an LDA dim of 14 and (2) on BABEL data with an LDA dim of 150. This PLDA is then used for adapting to the target languages. We used the score-averaging approach for the first PLDA, and multi-session scoring for the second PLDA system [13, 14]. The final scores are then fused with a weight of 0.5 for each PLDA system score.

4.2. Alternate system

The SVM system described in Section 3.1 is submitted as the alternate system.

5. EXPERIMENTS

In this section, we report our results on the test subset of LRE22 development set for all the described systems, along with the fusion results. We also report the time taken to process a single trial to generate a score vector.

As mentioned above, all systems are evaluated on the NIST's test split of the LRE 22 development set. The same test set is used to tune the fusion weights. The results–actual cost (act.C) and minimum cost (min.C)– as generated by the NIST scoring toolkit are presented in Table 4. The time required for processing a test segment to compute a score vector is given in Table 5. In each training condition, the results are presented for all described systems, and the results for submitted systems are labeled as Primary or Alternate.

For classifiers such as the RF and SVM, the score fusion does not significantly improve act.C, and SVM gives the lowest cost. In all training conditions for the PLDA classifier, fusion of the scores shows to improve the min.C and the act.C compared to using single systems. Although using a small subset of data to train a classifier provides the best performance, using large data to train PLDA classifiers provides better generalization capabilities. The effect of using large data for this task needs more investigation.

Table 5. Overview of the time required for processing a test segment. The time for fusing the scores is negligible compared to time taken by other modules. The timings were computed on a single-threaded Intel(R) Xeon(R) Gold 6248R based CPU machine with 8GB RAM.

System	Time (s)
Fixed Training	
Random Forest	2.98
Support Vector Machine	3.32
Fusion (**Primary**)	-
x-vector plda	0.76
speechbrain plda	2.90
Fusion (**Alternate**)	-
Open Training	
plda - LRE dev22	2.60
plda - BABEL	2.60
Fusion (**Primary**)	-
Support Vector Machine (**Alternate**)	3.3

6. ACKNOWLEDGEMENTS

This work was supported by the European Union's Horizon 2020 research and innovation program under grant agreement No. 833635 (project ROXANNE: Real-time network, text, and speaker analytics for combating organized crime, 2019-2022).

7. REFERENCES

[1] D. Povey *et al.*, "The kaldi speech recognition toolkit," in *IEEE workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.

[2] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. 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.

[3] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: emphasized channel attention, propagation and aggregation in TDNN based speaker verification," in *Interspeech 2020*, H. Meng, B. Xu, and T. F. Zheng, Eds. ISCA, 2020, pp. 3830–3834.

[4] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio, "SpeechBrain: A general-purpose speech toolkit," 2021, arXiv:2106.04624.

[5] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision*. Springer, 2006, pp. 531–542.

[6] J. Valk and T. Alumäe, "VoxLingua107: a dataset for spoken language recognition," in *Proc. IEEE SLT Workshop*, 2021.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss,

V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[8] I. Steinwart and A. Christmann, *Support vector machines*. Springer Science & Business Media, 2008.

[9] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[10] O. Kramer, "Scikit-learn," in *Machine learning for evolution strategies*. Springer, 2016, pp. 45–53.

[11] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[12] S. Madikeri, S. Dey, M. Ferras, P. Motlicek, and I. Himawan, "Idiap submission to the nist sre 2016 speaker recognition evaluation," Idiap, Tech. Rep., 2016.

[13] P. Rajan, A. Afanasyev, V. Hautamäki, and T. Kinnunen, "From single to multiple enrollment i-vectors: Practical plda scoring variants for speaker verification," *Digital Signal Processing*, vol. 31, pp. 93–101, 2014.

[14] S. Madikeri, M. Ferras, P. Motlicek, and S. Dey, "Intra-class covariance adaptation in plda back-ends for speaker verification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 5365–5369.

[15] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks." in *Interspeech*, 2018, pp. 3743–3747.