INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Advances in Language Recognition in Low Resource African Languages: The JHU-MIT Submission for NIST LRE22

Jesús Villalba[1,2], *Jonas Borgstrom*[3], *Maliha Jahan*[1], *Saurabh Kataria*[1,2], *L. Paola García-Perera*[1,2], *Pedro A. Torres-Carrasquillo*[3], *Najim Dehak*[1,2]

[1]Center for Language and Speech Processing, [2]Human Language Technology Center of Excellence, [1,2]Johns Hopkins University, Baltimore, MD, USA, [3]MIT Lincoln Laboratory, Lexington, MA, USA

jvillal7@jhu.edu, jonas.borgstrom@ll.mit.edu

Abstract

We present the effort of JHU-CLSP/HLTCOE and MIT Lincoln labs for NIST Language Recognition Evaluation (LRE) 2022. LRE22 consisted of a language detection task, i.e., determining whether a given target language was spoken in a speech segment. LRE22 focused on telephone and broadcast narrowband speech in African languages. Since LRE17, there has been large progress in neural embeddings, combined or not, with self-supervised models like Wav2Vec2. Therefore, one of our goals was to investigate these new models, i.e., ECAPA-TDNN, Res2Net, or Wav2Vec2+ECAPA-TDNN, in the LRE scenario. In the fixed training condition, LRE22 target languages were only included in a small development set. Hence, we focused on tuning our models to exploit the limited data. For the open condition, we built a massive training set including African data, which improved Cprimary by 50% w.r.t. fixed. Wav2Vec2 embeddings were the best, outperforming ECAPA and Res2Net by 11 and 3%, respectively.

Index Terms: language recognition, x-vectors, low-resource languages, African languages

1. Introduction

The National Institute of Standards and Technology (NIST) regularly performs language recognition evaluations (LRE) to appraise the state-of-the-art technology [1]. LREs consists of a language detection task, i.e., determining whether a given target language was spoken in a speech segment. As in previous evaluations, NIST LRE22[1] focused on conversational telephone speech (CTS) and broadcast narrowband speech (BNBS). Also, NIST LREs emphasized distinguishing between closely related languages. While LRE15-17 [2] had several language clusters, LRE22 entirely focused on African languages, mainly low-resource (Tunisian Arabic, Algerian Arabic, Libyan Arabic, North African French, Afrikaans, South African English, Indian South African English, Ndebele, Oromo, Tigrinya, Tsonga, Venda, Xhosa, and Zulu).

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited. This material is based upon work supported by the Department of Defense under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of Defense.

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

This paper presents the joint effort of JHU-CLSP/HLTCOE and MIT Lincoln labs for NIST LRE22. For LRE17, the best approaches were based on bottleneck features used as input to i-vector models or early TDNN x-vector networks [3, 4]. After these five years, there has been massive progress in neural embeddings, combined or not, with self-supervised models like Wav2Vec2 [5] or WavLM [6]. However, research in this area has been more intensive on the speaker recognition task rather than language. ECAPA-TDNN [7] and Res2Net [8] provided superior performance on VoxSRC [9] and NIST SRE [10] challenges. WavLM combined with ECAPA-TDNN outperformed previous approaches in the VoxCeleb benchmarks in [6]. Therefore, we investigated these new models, in the LRE scenario.

LRE22 offered fixed and open training conditions. For the fixed, LRE22 target languages were only included in a small development set. Hence, we focused on tuning our models to exploit the limited data. For the open, we built a massive training set of 150 languages, including the target African languages. The following sections describe our data setup, neural embeddings, back-ends, calibration, and fusion. We finalize discussing our evaluation results and conclusions.

2. Datasets

2.1. Training Fixed

The fixed condition data for neural embedding training were,

- **NIST LRE17 Train/Dev/Test:** It contains Arabic (Egyptian, Iraqi, Levantine, and Maghrebi), Chinese (Mandarin and Min Nan), English (British, American), Slavic (Polish, Russian), and Iberian Languages (Caribbean, European, and Latin American Spanish; and Brazilian Portuguese) [2] in 24.1k CTS and 4.4k audio from video (AfV) recordings.
- **Voxlingua107:** This is a language recognition dataset mined from Creative Commons (CC) YouTube videos [11]. This dataset contains 2.4M utterances from 107 languages. We removed Arabic, English, Portuguese, and Spanish languages because we want to discriminate between language variants, while Voxlingua only provides high-level labels.

In total, we obtained 2.5M segments from 118 languages.

2.2. Training Open

We added some extra datasets for the open condition, including Arabic dialects, South African, and Chinese languages.

- **NIST SRE CTS Superset:** This is a compilation of CTS data from NIST SRE04-12 [12]. We removed the files with undefined language; and generic English and Spanish labels. However, we kept US and Indian English. In total, we obtained 183.5k recordings from 25 languages.
- **NIST SRE16 dev/eval:** It contains 11.8k CTS recordings in Mandarin, Cantonese, Cebuano, and Tagalog languages [13].

10.21437/Interspeech.2023-1094

- **NIST SRE18-19 dev/eval:** It contains 32.5k CTS recordings in Tunisian Arabic [14, 15]
- **NIST SRE21:** It contains 16.6k CTS and 7.1k AfV in Mandarin, Cantonese, and Chinese-accented English [16].
- **IARPA Babel:** These are 154k CTS recordings in 13 languages (Assamese, Bengali, Georgian, Haitian, Kazakh, Kurmanji, Lao, Lithuanian, Pashto, Tamil, Telugu, Tok, Turkish, Vietnamese, and Zulu) from the IARPA babel program [17].
- **ADI17:** This is a dataset for fine-grained Arabic dialect identification collected from YouTube [18]. It contains more than 3k hours of data, 1M utterances, and 17 dialects.
- **FLEURS2022:** The Few-shot Learning Evaluation of Universal Representations of Speech [19] data consists of read speech using phrases from Wikipedia. We just used four languages included in the LRE22 (Afrikaans, Oromo, Xhosa, Zulu) with 12.2k segments.
- **Lwazi2009:** This corpus contains speech from the 11 official languages in South Africa [20]. We used seven languages (Afrikaans, Ndebele, South African English, Xitsonga, Tshivenda Xhosa, and Zulu) included in the LRE22 targets. Each language has 5 to 8 hours, about 200 speakers, and 30 utterances per speaker. The data is read and elicited speech recorded over landline or mobile channels.
- **NCHLT2014:** Wide-band speech from about 200 speakers per language in each of the eleven official languages of South Africa [21]. We used five languages (Afrikaans, Ndebele, Xitsonga, Xhosa, Zulu) with 50k segments.
- **AMMI2020:** Speech recordings were elicitated from text collected during the African Master of Machine Intelligence using mobile applications [22]. We used just the Tigrinya language with about 2.5h of speech.
- **AST2004:** A-law telephone speech simulating a hotel booking application collected for the African Speech Technology project [23]. We used Afrikaans variants, South African English, Indian South African English, Xhosa, and Zulu.
- **CommonVoice:** Multilingual read speech corpus [24] from which we got Indian English, Tigrinya, and French dialects (European, Canadian, and North African) (63.7k segments).

We combined the above datasets with the fixed condition data and removed VoxLingua107 French and LRE17 Maghrebi Arabic, obtaining 4M segments from 150 languages.

2.3. Development

LRE22 dev was used for training back-ends, performance evaluation, calibration, and fusion. We split it into two folds, so we trained a back-end in fold 1 and evaluated it in fold 2 and vice-versa. Then, we pooled the scores from both folds and trained a single calibration and fusion. Segments in LRE22 dev come from a small number of speakers. To have different speakers in each fold, we split the data as follows. First, an ECAPA-TDNN network computed language embeddings, and PCA reduced the embedding dimension while keeping 97.5% of the data variance. Then, for each language, we projected the embeddings to two dimensions using T-SNE and clustered them by Agglomerative Hierarchical Clustering (stopping threshold tuned by visualizing the T-SNE plots). Finally, each cluster was assigned to fold 1 or 2, so both folds have the same number of utterances.

2.4. LRE17 Maghrebi Arabic

LRE17 contains Maghrebi Arabic, while the LRE22 has Maghrebi sub-dialects (Tunisian, Algerian, and Libyan). To avoid pulling all Maghrebi dialects to the same embedding, we relabeled LRE17. First, we trained an ECAPA-TDNN on the fixed condition data with the original labels. Then, we trained a linear Gaussian classifier on the LRE22 dev Maghrebi dialects and evaluated it on LRE17. We kept the LRE17 segments with dialect posteriors larger than 0.975 and discarded the rest. For the open condition, we just discarded LRE17 since there were enough Maghrebi dialects in ADI17 and SRE18-19.

2.5. Augmentations

We applied telephone codecs to all non-telephone data. For large datasets (ADI, Voxlingua107, CommonVoice), we only used the version with codecs. For the rest, we used augmented and original audios. We used GSM, G711 mu/A-law, G222, G723.1, G726 (code sizes 2 to 5) and Opus (bitrates 4.5-32kbps).

For x-vector training, we augmented speech on the fly with MUSAN noise[2] and reverberation from RIR[3]. For back-end training, we augmented the LRE22 dev $10\times$ with different noise, reverberation, and random duration cuts (3-30 secs) and combined them with the original recording.

3. Neural Embeddings

We used language embedding architectures following the x-vector scheme [26]. The embedding network consists of an encoder that extracts frame-level discriminant embeddings, a pooling mechanism, and a classification head. We used ECAPA-TDNN and Res2Net architectures for the encoders, and channel-wise attentive statistics pooling [7]. For the open condition, we also used Wav2Vec2 encoder followed by ECAPA-TDNN. Unless indicated otherwise, the network minimizes additive angular margin softmax loss [27]. The acoustic features were 64 log-Mel filter-banks computed at 8 kHz sampling frequency. Features were short-time mean normalized with a 3 seconds window. Silence frames were removed using Kaldi energy VAD.

3.1. TSE-Res2Net50

Res2Net50 [8] consists of an input stem layer followed by 16 Res2Net bottleneck residual blocks. The bottleneck layer in the Res2Net blocks is divided into eight groups (scale). Each group (except the first one, which is just copied in the output) passes through a 3×3 convolution and is added to the input of the convolution of the next group. Hence, each group observes a different receptive field. We also used Time-Squeeze-Excitation (TSE) [28, 29], which scales channel and frequency dimensions at the output of each residual block according to their importance. The encoder output was reshaped from $(B, C, F/8, T/8)$–F =Mel filters, T =time– to $(B, C \times F/8, T/8)$ before the global pooling layer.

For the fixed condition, we trained on 3-second chunks and 512 effective batch-size–The actual batch size depended on GPU memory and network size, and gradient accumulation was used to achieve the desired effective batch size. We used Adam optimizer with lr=0.01 warmed up for 5k steps. After 40 steps, the learning rate was divided by two every 16k steps. We used AAM-Softmax [27] objective with margin=0.2 scale=30 and Inter-Top margin=0.1 with K=5 [30]. For the open condition, we halved the learning rate every 24k steps.

The networks were fine-tuned by uniformly sampling languages. We used SGD with lr=0.1 and momentum=0.9. The learning rate warmed up for 5k steps, then divided by two every 4k steps. We tried several margin values and chunk-sizes.

[2]http://www.openslr.org/resources/17

[3]http://www.openslr.org/resources/28

Table 1: *Fixed Condition Results on LRE22 dev and eval.*

System			LRE22 dev		LRE22 eval	
Name	Embed.	BE	Min Cp	Act Cp	Min Cp	Act Cp
NIST Baseline [25]					0.600	0.730
1	ECAPA-TDNN	Linear GBE	0.257	0.259	0.254	0.256
2		Linear SVM	0.266	0.268	0.279	0.282
3		GCA	0.245	0.247	0.270	0.270
4		Gauss. SVM	0.205	0.207	0.278	0.280
5	TSE-Res2Net	Linear GBE	0.248	0.248	0.251	0.252
6		Linear SVM	0.239	0.242	0.273	0.274
7		GCA	0.232	0.235	0.270	0.270
8 Single		Gauss. SVM	**0.183**	**0.186**	0.282	0.286
9	ECAPA-TDNN-Focal	Linear GBE	0.239	0.243	**0.246**	**0.247**
10		Linear SVM	0.245	0.246	0.274	0.274
11		Gauss. SVM	0.188	0.190	0.279	0.282
Eval						
7-sys	3+4+7-11		**0.153**	0.156	**0.220**	**0.224**
Primary 6-sys	3+4+7+8+10+11		**0.153**	**0.155**	**0.220**	**0.224**
5-sys	3+7+8+10+11		**0.154**	**0.155**	0.221	0.225
4-sys	7+8+10+11		**0.153**	**0.154**	0.223	0.227
3-sys	7+8+11		0.156	0.159	0.227	0.233
2-sys	8+11		0.162	0.163	0.250	0.254
Post-Eval						
Post 1	1+3+5+7+9		**0.197**	0.200	0.213	0.214
Post 2	3+7+9		**0.197**	**0.198**	**0.212**	**0.213**
Post 3	1+3+9		0.218	0.219	0.218	0.221
Post 4	7+9		0.203	0.205	0.216	0.216

We also switched the loss function to Subcenter AAM-Softmax with two subcenters [31]. We concluded that fine-tuning with long recordings or large margins hurt–large margin could raise Cprimary $\times 2$, so we set the chunk back to 3 seconds and margin=0. The number of epochs in each stage was selected by evaluating the model epoch by epoch on the dev data. We also tried fine-tuning with hard prototype mining [32] without improvements.

3.2. ECAPA-TDNN

ECAPA-TDNN [7] can be regarded as a TSE-Res2Net with 1D dilated convolutions. Following [9], we used a network with four Res2Net blocks with 2048 channels each. The outputs of each Res2Net block were concatenated at the encoder output and projected to 4096 dimensions before global pooling.

The network was trained following the same scheme as the TSE-Res2Net. In this case, we obtained a slight improvement by a second finetuning stage with hard prototype mining with eight hard prototypes per language. We also fine-tuned another network with Focal loss [33] instead of AAM-softmax, Cutmix regularization [34], and 16 hard prototypes.

3.3. Wav2Vec2 + ECAPA-TDNN

This extractor follows the scheme in [6]. The input audio was upsampled to 16 kHz and processed by a Multilingual Wav2Vec2 with 300M params trained on 128 languages[4] [5]. We compute a weighted average of the transformer's hidden layers. Finally, we fed the result into an ECAPA-TDNN with 3 Res2Net layers and 1024 neurons per layer.

The ECAPA-TDNN and transformer layers' weights were trained while the Wav2Vec2 encoder was frozen. The loss function was Subcenter AAM-Softmax with scale$=32$, margin$=0$, and two subcenters. We used SGD optimizer with a maximum learning rate of 0.45, momentum=0.9, and effective batch-size=1024. The learning rate was warmed up for 5k steps. After 32k steps, we halved the learning rate every 16k steps. We trained using 3-second chunks for nine epochs. We also tried to finetune the wav2vec2 encoder, but it over-fitted damaging performance.

[4]https://huggingface.co/facebook/wav2vec2-xls-r-300m

4. Back-ends

The back-ends were mainly trained on LRE22 dev augmented as indicated in Section 2.5. To score the LRE22 dev, we divided the dev data into two folds as explained in Section 2.3, trained on fold 1 and evaluated on fold 2 and vice-versa. To score the LRE22 eval, we trained the back-end on the full LRE22 dev data. All back-ends preprocessed the data by centering, whitening, and length normalization.

4.1. Gaussian Back-end

This was a linear Gaussian classifier with one Gaussian per target language and a covariance shared across languages.

4.2. Linear/Gaussian SVM

These were multi-class SVM classifiers with linear or Gaussian kernels. They were trained on LRE22 dev augmented and LRE17. LRE17 Maghrebi Arabic was relabeled into the three target Arabic dialects as explained in Section 2.4. The remaining languages in LRE17 were added as negative samples.

4.3. Generative Condition-Aware (GCA) Back-end

The GCA back-end jointly models embeddings and their associated vectors of language log-likelihoods extracted using a Gaussian back-end classifier. The model assumes a discrete set of underlying signal conditions, modeled as latent random variables, which can represent acoustic conditions in the input signals due to extrinsic variabilities. Embeddings and score vectors are then modeled as condition-dependent Normal variables. The Expectation-Maximization (EM) algorithm was used for parameter estimation. During inference, we marginalize over the latent conditions. The GCA back-end extends the generative condition-aware score calibration proposed in [35].

Table 2: *Open Condition Results on LRE22 dev and eval.*

System			LRE22 dev		LRE22 eval	
Name	Embed.	BE	Min Cp	Act Cp	Min Cp	Act Cp
1	ECAPA-TDNN	Linear GBE	0.125	0.130	0.144	0.144
2		Linear SVM	0.129	0.129	0.172	0.172
3		GCA	0.119	0.121	0.150	0.150
4		Gauss. SVM	0.090	0.091	0.160	0.161
5	TSE-Res2Net	Linear GBE	0.105	0.107	0.126	0.128
6		Linear SVM	0.126	0.128	0.154	0.154
7		GCA	0.104	0.104	0.136	0.136
8 Alt. Single		Gauss. SVM	**0.092**	**0.093**	0.142	0.143
9	Wav2Vec2	Linear GBE	0.094	0.095	**0.123**	**0.124**
10		Linear SVM	0.112	0.113	0.144	0.144
11 Single		Gauss. SVM	**0.088**	**0.089**	0.140	0.141
Eval						
Primary 6-sys	3+4+7+8+9+11		**0.055**	**0.056**	**0.094**	**0.095**
2 nets $\times$ 2 be	7+8+9+11		0.056	0.057	0.098	0.099
3 nets $\times$ 1 be	4+7+11		0.056	0.058	0.097	0.098
2 nets $\times$ 1 be	7+11		0.060	0.061	0.102	0.103
Post-Eval						
Post 1	1+3+5+7+9		**0.062**	**0.064**	**0.089**	0.090
Post 2	3+5+7+9		0.065	0.066	**0.089**	**0.089**
Post 3	3+7+9		0.065	0.065	**0.089**	0.090
Post 4	7+9		0.065	0.066	0.096	0.096
Post 5	1+5+9		0.071	0.073	0.094	0.094

5. Calibration and Fusion

Linear logistic regression with the Focal Multiclass toolkit[5] was used for calibration and fusion. They were trained on the 2-fold cross-validation LRE22 dev scores and applied on dev and eval. First, we calibrated each system separately. Then, we fused them on top of the calibrated scores. This allowed us to obtain meaningful fusion weights. We started fusing all embeddings networks $\times$ all back-ends. Based on the weights, we decided which systems to remove to obtain fusions of different sizes.

6. Results

Tables 1 and 2 show the results for our fixed and open condition single systems and fusions.

6.1. Single Systems

For the fixed condition, TSE-Res2NET and ECAPA-TDNN finetuned with focal loss were comparable in dev and better than ECAPA-TDNN finetuned with AM-softmax loss. However, the distance between networks was reduced in eval, being all comparable. For the open condition, Wav2Vec2-based embeddings were the best in dev and eval, closely followed by TSE-Res2Net, and at a significant distance from ECAPA-TDNN. This result suggests that we need to train the embeddings on data that include the target languages to take advantage of the capacity of the most complex architectures.

Regarding back-ends, Gauss SVM was significantly better on dev but over-fitted, being the worst on eval despite our clustering method to assign different speakers to each dev fold. GCA also over-fitted but to a lesser degree. The most simple linear Gaussian back-end was the best on eval.

6.2. Fusions

Primary fusions combined six systems (3 embeddings $\times$ 2 back-ends). They improved Cprimary by 21 and 32% relative w.r.t. the best single system in the fixed and open conditions, respectively. Improvements by fusing two systems were 11 and 26%, for fixed and open. Open condition fusions improved 50-56% w.r.t. their fixed counterparts. For individual languages, open condition relative improvement ranged from 26% (Eng-IAF, Fra-NTF) to 75% (Oromo).

In post-eval, we repeated the fusions removing the over-fitted SVM back-ends obtaining a 5% relative improvement in the eval set. In this case, the gain of fusing five systems was reduced to just 6%. This result indicates that fusion helped to counteract the negative effect of over-fitted back-ends in our original submissions. However, the fusion gains were minimal when using more robust back-ends, indicating that all embeddings captured similar information. Despite GCA being slightly over-fitted, fusions combining GBE and GCA back-ends performed better than fusions using just the GBE back-end–compare Post 3 vs Post 5 in Table 2.

7. Conclusions

Our submissions to NIST LRE22 were fusions of different embedding networks combined with different Gaussian and SVM back-ends. In the fixed training condition, we observed little difference between network architectures. Meanwhile, for the open condition, more powerful architectures (Res2Net, Wav2Vec2) outperformed smaller networks (ECAPA-TDNN). This indicates that larger architectures must be trained on target language data to be effective. Our best single system was a multilingual Wav2Vec2 model pre-trained on 128 languages with an ECAPA-TDNN head trained on our open dataset of 150 languages. Regarding back-ends, the simple linear Gaussian back-end performed the best in eval while SVMs over-fitted. Removing SVM back-ends from the fusion improved Cprimary by 6%. We created an open-condition training set comprising 12 extra datasets containing LRE22 target languages and others. In total, we gathered 4M recordings from 150 languages. This provided a 50% improvement w.r.t. fixed condition. We will release recipes to replicate our results at publication time[6].

[5] https://sites.google.com/site/nikobrummer/focalmulticlass

[6] https://github.com/hyperion-ml/hyperion/tree/master/egs/lre22

8. References

[1] A. F. Martin, C. S. Greenberg, J. M. Howard, G. R. Doddington, and J. J. Godfrey, "NIST Language Recognition Evaluation - Past and Future," in *Proc. The Speaker and Language Recognition Workshop (Odyssey 2014)*, 2014, pp. 145–151.

[2] S. O. Sadjadi, T. Kheyrkhah, C. Greenberg, E. Singer, D. Reynolds, L. Mason, and J. Hernandez-Cordero, "Performance Analysis of the 2017 NIST Language Recognition Evaluation," in *Proc. Interspeech 2018*, 2018, pp. 1798–1802.

[3] F. Richardson, P. A. Torres-Carrasquillo, J. Borgstrom, D. Sturim, Y. Gwon, J. Villalba, N. Chen, J. Trmal, N. Chen, and N. Dehak, "The MIT Lincoln Laboratory / JHU / EPITA-LSE LRE17 System," in *Proceedings of Odyssey 2018 - The Speaker and Language Recognition Workshop*, Les Sables d'Olonne, France, 2018.

[4] A. Mccree, D. Snyder, G. Sell, and D. Garcia-Romero, "Language Recognition for Telephone and Video Speech: The JHU HLTCOE Submission for NIST LRE17 ," in *The Speaker and Language Recognition Workshop (Odyssey 2018)*, 2018, pp. 68–73.

[5] A. Babu, C. Wang, A. Tjandra, K. Lakhotia, Q. Xu, N. Goyal, K. Singh, P. von Platen, Y. Saraf, J. Pino, A. Baevski, A. Conneau, and M. Auli, "XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale," in *Proc. Interspeech 2022*, 2022, pp. 2278–2282.

[6] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, et al., "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[7] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification," in *Interspeech 2020*, 2020.

[8] S. Gao, M. Cheng, K. Zhao, X. Zhang, M. Yang, and P. Torr, "Res2Net: A New Multi-Scale Backbone Architecture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 2, pp. 652–662, feb 2021.

[9] J. Thienpondt, B. Desplanques, and K. Demuynck, "The Idlab VoxSRC-20 Submission: Large Margin Fine-Tuning and Quality-Aware Score Calibration in DNN Based Speaker Verification," in *ICASSP 2021*. IEEE, 2021, pp. 5814–5818.

[10] J. Villalba, B. J Borgstrom, S. Kataria, M. Rybicka, C. D. Castillo, J. Cho, L. P. García-Perera, P. A. Torres-Carrasquillo, and N. Dehak, "Advances in Cross-Lingual and Cross-Source Audio-Visual Speaker Recognition: The JHU-MIT System for NIST SRE21," 6 2022, pp. 213–220, ISCA.

[11] J. Valk and T. Alumäe, "VoxLingua107: a Dataset for Spoken Language Recognition," in *Proc. IEEE SLT Workshop*, 2021.

[12] S. O. Sadjadi, "NIST SRE CTS Superset: A large-scale dataset for telephony speaker recognition," *arXiv preprint arXiv:2108.07118*, 2021.

[13] S. O. Sadjadi, T. Kheyrkhah, A. Tong, C. Greenberg, D. Reynolds, E. Singer, L. Mason, and J. Hernandez-Cordero, "The 2016 NIST Speaker Recognition Evaluation," 8 2017, pp. 1353–1357, ISCA.

[14] S. O. Sadjadi, C. S. Greenberg, D. A. Reynolds, E. Singer, L. Mason, and J. Hernandez-Cordero, "The 2018 NIST speaker recognition evaluation," 8 2019, pp. 1483–1487.

[15] S. O. Sadjadi, C. Greenberg, E. Singer, D. Reynolds, L. Mason, and J. Hernandez-Cordero, "The 2019 NIST Speaker Recognition Evaluation CTS Challenge," 11 2020, pp. 266–272, ISCA.

[16] O. Sadjadi, C. Greenberg, E. Singer, L. Mason, D. Reynolds, et al., "NIST 2021 speaker recognition evaluation plan," 2021.

[17] M. Harper, "Learning from 26 languages: Program management and science in the babel program," in *COLING 2014*, 2014.

[18] S. Shon, A. Ali, Y. Samih, H. Mubarak, and J. Glass, "Adi17: A fine-grained arabic dialect identification dataset," 5 2020, pp. 8244–8248, IEEE.

[19] A. Conneau, M. Ma, S. Khanuja, Y. Zhang, V. Axelrod, S. Dalmia, J. Riesa, C. Rivera, and A. Bapna, "Fleurs: Few-shot learning evaluation of universal representations of speech," in *IEEE Spoken Language Technology Workshop (SLT)*, 2022, pp. 798–805.

[20] E. Barnard, M. Davel, and C. Van Heerden, "Asr corpus design for resource-scarce languages," Interspeech 2009, 2009.

[21] E. Barnard, M. H. Davel, C. van Heerden, F. De Wet, and J. Badenhorst, "The nchlt speech corpus of the south african languages," Workshop Spoken Language Technologies for Under-resourced Languages (SLTU), 2014.

[22] J. H. Mohamud, L. A. Thompson, A. Ndoye, and L. Besacier, "Fast development of asr in african languages using self supervised speech representation learning," *arXiv preprint arXiv:2103.08993*, 2021.

[23] J. C. Roux, P. H. Louw, and T. R. Niesler, "The African speech technology project: An assessment," in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, May 2004.

[24] R. Ardila, M. Branson, K. Davis, M. Kohler, J. Meyer, M. Henretty, R. Morais, L. Saunders, F. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 4218–4222.

[25] Y. Lee, C. Greenberg, E. Godard, A. Butt, E. Singer, T. Nguyen, L. Mason, and D. Reynolds, "The 2022 NIST Language Recognition Evaluation," in *Interspeech 2023*, Dublin, Ireland, aug 2023, pp. 496–500, ISCA.

[26] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors : Robust DNN Embeddings for Speaker Recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, Alberta, Canada, apr 2018, pp. 5329–5333, IEEE.

[27] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4685–4694.

[28] S. Kataria, P. S. Nidadavolu, J. Villalba, N. Chen, L. P. Garcia-Perera, and N. Dehak, "Feature Enhancement with Deep Feature Losses for Speaker Verification," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, may 2020, pp. 7584–7588.

[29] M. Rybicka, J. Villalba, P. Żelasko, N. Dehak, and K. Kowalczyk, "Spine2Net: SpineNet with Res2Net and Time-Squeeze-and-Excitation Blocks for Speaker Recognition," in *Interspeech 2021*, Brno, Czech Republic, aug 2021, pp. 496–500, ISCA.

[30] M. Zhao, Y. Ma, Y. Ding, Y. Zheng, M. Liu, and M. Xu, "Multi-query multi-head attention pooling and inter-topk penalty for speaker verification," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6737–6741.

[31] J. Deng, J. Guo, T. Liu, M. Gong, and S. Zafeiriou, "Sub-center ArcFace: Boosting Face Recognition by Large-Scale Noisy Web Faces," 10 2020, pp. 741–757.

[32] J. Thienpondt, B. Desplanques, and K. Demuynck, "Cross-Lingual Speaker Verification with Domain-Balanced Hard Prototype Mining and Language-Dependent Score Normalization," in *Proc. Interspeech 2020*, 2020, pp. 756–760.

[33] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[34] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6023–6032.

[35] B. J. Borgstrom, "A generative approach to condition-aware score calibration for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, 2022.