Description and analysis of the KPT system for NIST Language Recognition Evaluation 2022

INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Description and analysis of the KPT system for NIST Language Recognition Evaluation 2022

Salvatore Sarni[1], Sandro Cumani[1], Sabato Marco Siniscalchi[2], Andrea Bottino[1]

[1]Politecnico di Torino, Italy
[2]Kore University of Enna, Italy

salvatore.sarni@polito.it, sandro.cumani@polito.it, marco.siniscalchi@unikore.it, andrea.bottino@polito.it

Abstract

This paper presents an analysis of the KPT system for the 2022 NIST Language Recognition Evaluation. The KPT submission focuses on the fixed training condition where only specific speech data can be used to develop all the modules and auxiliary systems used to build the language recognizer. Our solution consists of several sub-systems based on different neural network front-ends and a common back-end for classification and fusion. The goal of each front-end is to extract language-related embeddings. Gaussian linear models are used to classify the embeddings of each front-end, followed by multi-class logistic regression to calibrate and fuse the different sub-systems. Experimental results from the NIST LRE 2022 evaluation task show that our approach achieves competitive performance.
Index Terms: language recognition evaluation, language embeddings, deep neural networks, language classification

1. Introduction

NIST 2022 Language Recognition Evaluation (LRE22) [1] is the latest in a series of evaluations promoted by NIST to assess the current state of the art in language recognition. The LRE22 task consists of a language detection problem in which it is necessary to determine whether a specific language from 14 alternatives was spoken in a particular utterance. The main task consists of a fixed training data condition that allows only certain datasets to be used for training the recognition systems. The KPT submission combines the efforts of Politecnico di Torino and Kore University of Enna for LRE22. Our system is based on the fusion of 15 sub-systems that use six different deep neural network (DNN) architectures, described in Section 3, as embedding extractors. Some sub-systems use the same architecture but are trained for a different number of epochs. This solution arises from preliminary results showing the effectiveness of merging these variants at score level. The same classification back-end, based on a Gaussian Linear Classifier (GLC) [2], was used for all sub-systems. The rationale for this choice stems from our preliminary evaluation, where we assessed several back-ends, namely: (i) Logistic Regression, (ii) Support Vector Machines, (iii) Pairwise Support Vector Machines [3, 4], (iv) Probabilistic Linear Discriminant Analysis (PLDA) [5], and (v) their duration-aware variants [6, 7, 8]. Although some of these classifiers (especially PLDA and its extensions) were able to improve performance over GLC for individual sub-systems, they did not offer advantages over a pure GLC setup when multiple sub-systems were combined. Calibration of each sub-system and final fusion were based on linear multi-class logistic regression [9]. Given the limited amount of LRE22 development data, care was devoted to devise a training protocol that would allow the majority of the data to be used for training both the classification back-ends and the fusion model. Our results show that our procedure, based on an ad-hoc Leave-One-Out scheme, achieves very good performance that is consistent across development and evaluation data. We also analyze the contribution of the different models and show that the large number of systems did not lead to overfitting, despite the limited amount of training data for target languages. For a comparative analysis of our results with those of other participants, confirming the effectiveness of our approach, we refer the reader to [10], where our submission is identified as team "T2".[1]

The paper is organized as follows. Section 2 briefly describes the LRE22 task and the fixed condition data. Section 3 details our embedding extractors, while Section 4 introduces our back-end and fusion strategies. The results and analysis of the contributions of the different building blocks can be found in Section 5. Finally, we present our conclusions in Section 6.

2. NIST 2022 Language Recognition Evaluation

The LRE22 fixed condition training set consists of NIST LRE 2017 (LRE17) training and test data [11], VoxLingua-107 (VOX) [12] data and the LRE22 Development set. Among these, only the latter set contains labeled segments belonging to the 14 target languages. It consists of 30 audio segments per language, which have been subdivided into 10 sub-segments per audio segment by NIST, resulting in a total of 300 audio files per language with nominal durations ranging from 3 to 93 seconds.

3. Front-ends

Given the limited amount of data for the target languages, the LRE22 Development set was used only for training the back-end and fusion models. The embedding extractors were instead trained on the LRE17 training data and the VOX datasets. These data were organized into two different lists. The first consists of all VOX data. The second (VOX + LRE) consists of all VOX languages not included in the LRE17 set and the LRE17 training data. The reason for removing a subset of VOX languages is that LRE17 contains dialects of different languages that are present in VOX only as a single class. The evaluation part of LRE17 was withheld for internal validation.

3.1. Acoustic features

The embedding extractors were trained with the following two sets of acoustic features:

[1]Due to evaluation rules [1], we cannot directly publish comparative results with other teams. The work [10] provides the official, anonymized results disclosed by the evaluation organizers.

10.21437/Interspeech.2023-155

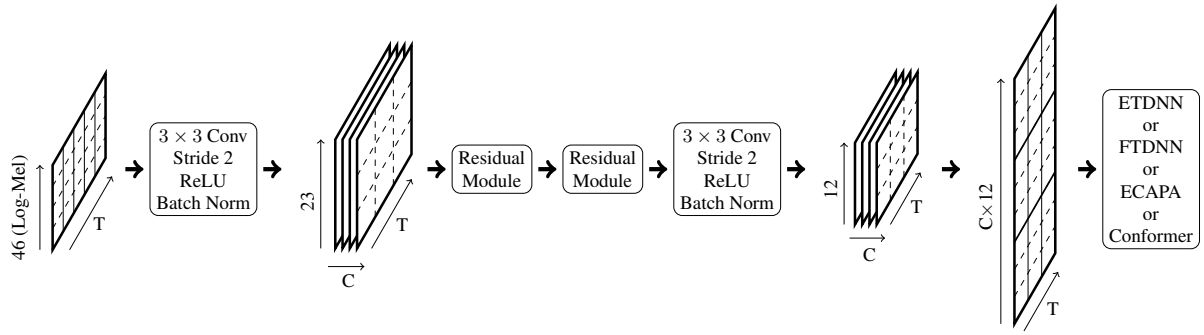

Figure 1: *CNN block for CNN-based networks. The 2D CNN blocks transform the acoustic features in a set of higher-level features that are then fed to the different deep neural networks.*

46 Log-Mel: we extracted 46 Log-Mel band parameters with short time centering (STC) computed over both speech and non-speech audio frames. The speech frames were extracted using energy-based voice activity detection (VAD).

23 MFCC: we extracted 23-dimensional MFCCs with a frame length of 25ms, which were mean-normalized over a sliding window. An energy-based VAD was used to filter out non-speech frames.

3.2. Embedding extractors

Most embedding extractors employ custom implementations of known architectures. Unless otherwise noted, networks were trained with 3-second segments and a batch size of 128. We used Stochastic Gradient Descent (SGD) as the optimizer with a momentum of 0.9 and a weight decay of 10^{-4}. A cyclic learning rate approach [13] was adopted, with 10^{-5} as the starting value and 10^{-1} as the maximum value. At each epoch corresponding to one iteration over all segments of the entire training set, the maximum LR is halved. In our initial experiments, we found that for some extractors, very few epochs are sufficient to achieve optimal performance. Therefore, to select the most promising models, we divided each epoch into 10 mini-epochs and evaluated the performance of the different networks as a function of the number of mini-epochs.

In the following, all models whose name starts with CNN contain an initial module based on 2D convolutions inspired by [14]. This module, shown in Figure 1, consists of a 2D convolutional layer with a stride of 2, followed by two 2D convolutional residual blocks, and a final 2D convolutional layer with a stride of also 2. All layers have a channel size of 128 and a kernel of size 3. The channel and frequency dimensions of the output are then flattened to one dimension and used as input for the different architectures.

3.2.1. CNN-ETDNN

The Extended Time Delay Neural Network (ETDNN) [15] is the extension of the classical x-vector model introduced in [16]. The network is implemented as in [17], with the addition of the CNN module mentioned above. The network is trained with a cross-entropy loss. The resulting embeddings are the output of the second last fully-connected layer and have a size of 512.

3.2.2. CNN-ECAPA

The ECAPA architecture is implemented according to [18], with the channel dimension set to 512. The network was trained using Additive Angular Margin softmax [19, 20] with a margin of 0.2 and a scale factor of 30. The 192-dimensional embeddings are extracted from the last fully-connected layer.

3.2.3. CNN-Factorized TDNN

The Factorized TDNN (FTDNN) [21] introduces a factorization of the weight matrix of each layer, with one of the factors constrained to be semi-orthogonal. We adopt the "floating" constraint of [21]. Our implementation follows the details in [17], and the network is trained with cross-entropy loss. The 512-dimensional embeddings are extracted from the last fully-connected layer after the pooling layer.

3.2.4. Conformer

The implementation of the Conformer architecture follows [22], with the sub-sampling module replaced by a 1-dimensional convolutional block. The network was trained with Additive Margin softmax [23, 20] with a margin of 0.2 and a scale factor of 30. The 192-dimensional embeddings are extracted from the last fully-connected layer.

3.2.5. TDNN

The time-delay neural model (TDNN) is the building block of the x-vector architecture used to extract the 512-dimensional language embeddings according to the implementation in [16]. The embeddings are extracted from the second last fully-connected layer. The model is trained with a natural gradient extension of SGD with momentum [24]. The learning rate starts at 0.01 and ends at 0.001 and is decreased by a factor of 10 during training with an exponential schedule. For more details on the training phase, the reader is referred to [24].

3.2.6. MagNetO

The MagNetO architecture is implemented as in [25]. The network was trained with Additive Margin softmax [23, 20] with a margin of 0.2 and a scale factor of 40. 512-dimensional embeddings are extracted from the last fully-connected layer. SGD was used for training, with a momentum of 0.9 and a weight decay of $5e^{-5}$. An exponentially decaying learning rate was used during training. The starting LR is 10^{-1} and is halved at each epoch.

The set of networks, selected for the final submission based on fusion results on the development set, is given in Table 1. In the same table, we also report the model size of each network.

Table 1: *Additional details and results for the different sub-systems of the KPT primary submission.*

ID	DNN	Params	Mini epochs†	Emb. Size	PCA	Act C_{prim} — Dev†† Sub-System	Act C_{prim} — Dev†† Net Fusion	Act C_{prim} — Eval Sub-System	Act C_{prim} — Eval Net Fusion
Training Set: VOX + LRE — Acoustic features: 46 Log-Mel									
1	CNN-ECAPA	10.7M	1	192	—	0.337	0.232	0.353	0.239
2	CNN-ECAPA		2		150	0.304		0.314	
3	CNN-ECAPA		10		150	0.268		0.269	
4	CNN-FTDNN	18.5M	7	512	200	0.256	0.202	0.280	0.210
5	CNN-FTDNN		20		200	0.233		0.235	
6	CNN-FTDNN		60		200	0.227		0.232	
7	CNN-FTDNN		70		200	0.234		0.234	
8	CNN-FTDNN		100		200	0.243		0.240	
9	CNN-ETDNN	10.7M	50	512	200	0.206	0.202	0.226	0.219
10	CNN-ETDNN		70		200	0.210		0.224	
11	Conformer	18.4M	10	192	—	0.340	0.302	0.338	0.301
12	Conformer		30		—	0.350		0.336	
13	Conformer		40		—	0.336		0.343	
14	MagNetO	28.5M	30	512	100	0.278	0.278	0.301	0.301
Training Set: VOX — Acoustic features: 23 MFCC									
15	TDNN	4.5M	200	512	150	0.325	0.325	0.320	0.320
Primary system (Fusion of all sub-systems)						**0.161**		**0.187**	

† One epoch (full training set) corresponds to 10 mini-epochs
†† Development results are computed using the LOO approach detailed in Section 4

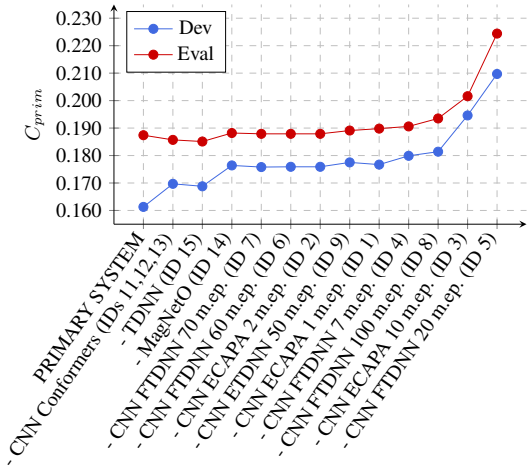

Figure 2: *Performance of different system fusions on the development and evaluation sets. The left-most column (PRIMARY SYSTEM) refers to the primary submission, that includes all systems. Left-to-right are the results obtained by* incrementally *removing the least contributing sub-system(s), shown on the x-axis, based on evaluation results. The right-most column corresponds to sub-system ID 10.*

4. Back-end, calibration and fusion

The VOXLingua-107 and LRE17 datasets do not contain data for most of the target languages of LRE22. Therefore, we used the LRE22 Development set to train the back-end, calibration, and fusion models as well as for internal evaluation and model selection. A Gaussian Linear Classifier (GLC) [2] was trained over the embeddings of each front-end. The embeddings were pre-processed using Principal Component Analysis (PCA), whose dimensionality was chosen to optimize the results of the fused system rather than those of each individual sub-system. The PCA dimensions for each sub-system are given in Table 1. Calibration is based on multi-class logistic regression [9] trained to compute a linear mapping of the classification scores to a set of class-conditional log-likelihoods[2]. The model consists of a single scalar and a set of language-dependent bias terms. Fusion follows a similar strategy, with a linear model containing a single scalar per sub-system and a bias vector for each language. In order to use the entire LRE22-Dev set for training the back-end, calibration, and fusion models, as well as for evaluating the different systems, we used the following leave-one-out (LOO) procedure.

- For each language, we group the segments belonging to the same audio file (30 groups per language).
- For each language, we randomly shuffle the groups and then iteratively remove one group of segments (for a total of 14 groups per iteration, corresponding to 140 segments), while using the remaining 29 for training the back-end classifier. At each iteration i, the 140 removed segments are then scored with the trained classifier $\mathcal{M}_i$. The scores obtained by each classifier are then pooled together to form the dataset of scores used to train the fusion models. An additional model $\mathcal{M}_P$ is trained over the entire development set. This model

[2]Up to an utterance-dependent but language-independent bias that is irrelevant to compute likelihood ratios.

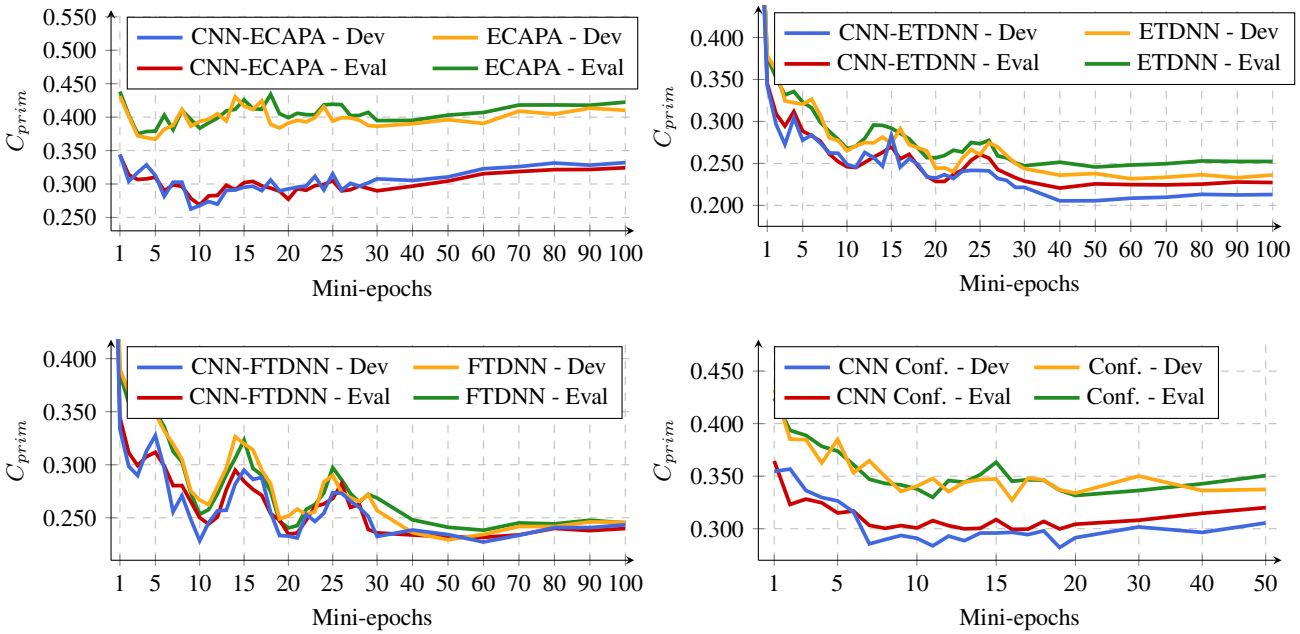

Figure 3: *Performance as a function of mini-epochs for different architectures (Top-Left: ECAPA, Top-Right: ETDNN, Bottom-Left: FTDNN, Bottom-Right: Conformer).*

is then used to score the evaluation segments.

- We repeat the LOO scheme starting from the raw pooled scores created in the previous step. The groups are randomly shuffled again, using a different seed. At each iteration, a calibration model $\mathcal{C}_i$ or a fusion model $\mathcal{F}_i$ is trained for each sub-system or for a combination of sub-systems and applied to the scores of the left-out segments. The calibrated / fused scores are then pooled to compute the Dev metrics of Table 1. The final fusion model $\mathcal{F}_P$ is trained over the entire set of raw pooled scores and applied to the evaluation scores generated by the back-end $\mathcal{M}_P$.

5. Results

Table 1 reports the actual primary cost (C_{prim}) for each sub-system on the LRE22 evaluation set and on the LRE22 development set using the leave-one-out procedure described in Section 4. For each DNN architecture, we also report the performance of partial fusions (Net Fusion column) that combine models with the same architecture trained with a different number of epochs. We can observe that for most architectures the scores of models trained with different numbers of epochs provide complementary information and that their fusion improves the results in both the evaluation and development sets. Finally, the last row of the table gives the actual costs for our primary submission (fusion of all individual sub-systems), which achieved competitive results for the task, as shown in [10], where our team submission is labeled as "T2".

To assess the contribution of each sub-system to the primary fusion, we analyzed the effects of successively removing individual sub-systems from our primary fusion. At each step, we chose to incrementally remove the system that would cause the least degradation or the best improvement of the actual C_{prim} in the evaluation set. The results are shown in Figure 2. The left-most column corresponds to the fusion of all sub-systems, while from left to right we report the results obtained by incrementally removing the sub-system(s) with the lowest contribution. We can observe that (i) our primary submission achieves close to optimal results, (ii) similar results could have been obtained with a much smaller set of sub-systems, but (iii) the large number of systems does not cause significant overfitting and the results on development and evaluation data are mostly consistent, confirming the strength of our training protocol.

Finally, Figure 3 shows the performance of the different front-ends as a function of mini-epochs ($1/10$ of an epoch) and the contribution of the 2D convolutional input blocks described in Section 3.2. For most architectures, good results can be obtained with a small number of mini-epochs. For ECAPA and conformers, training for multiple epochs actually leads to significant overfitting. Even in this case, the development and evaluation results are consistent. We can also observe that the 2D convolutional input blocks consistently improve performance for all models.

6. Conclusions

We presented the KPT submission for the NIST 2022 Language Recognition Evaluation. The primary system was based on the fusion of multiple sub-systems using different neural network front-ends for embedding extraction and a GLC classification back-end. The fusion was implemented through multi-class logistic regression. We analyzed the impact of our main contributions, a robust protocol for training the back-end, the extension of DNN embeddings with a 2D CNN input block, and aggressive early stopping for DNN training, showing that our approach was indeed effective for the task.

7. Acknowledgments

We would like to thank Prof. Pietro Laface for preliminary discussions and his support to our work, and Daniele Colibro and Claudio Vair for useful discussions on NIST evaluations.

8. References

[1] "NIST 2022 language recognition evaluation plan," 2022, available at https://lre.nist.gov/uassets/3.

[2] G. Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matějka, "Language recognition in i-vectors space," in *Proceedings of Interspeech 2011*, 2011, pp. 861–864.

[3] S. Cumani, N. Brümmer, L. Burget, P. Laface, O. Plchot, and V. Vasilakakis, "Pairwise discriminative speaker verification in the i-vector space," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1217–1227, 2013.

[4] S. Cumani and P. Laface, "Large scale training of Pairwise Support Vector Machines for speaker recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 11, pp. 1590–1600, 2014.

[5] S. Ioffe, "Probabilistic linear discriminant analysis," in *Proceedings of the 9th European Conference on Computer Vision*, ser. ECCV'06, vol. Part IV, 2006, pp. 531–542.

[6] S. Cumani, O. Plchot, and R. Fér, "Exploiting i-vector posterior covariances for short-duration language recognition," in *INTERSPEECH*, 2015.

[7] S. Cumani, O. Plchot, and P. Laface, "On the use of i-vector posterior distributions in Probabilistic Linear Discriminant Analysis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 846–857, 2014.

[8] S. Cumani, "Fast scoring of full posterior PLDA models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 2036–2045, 2015.

[9] N. Brummer, S. Cumani, O. Glembek, M. Karafiát, P. Matejka, J. Pesan, O. Plchot, M. Soufifar, E. Villiers, and J. Černocký, "Description and analysis of the brno 276 system for lre2011," in *Odyssey 2012: The Speaker and Language RecognitionWorkshop*, 2012, pp. 216–223.

[10] Y. Lee, C. Greenberg, E. Godard, A. A. Butt, E. Singer, T. Nguyen, L. Mason, and D. Reynolds, "The 2022 nist language recognition evaluation," 2023. [Online]. Available: https://arxiv.org/abs/2302.14624

[11] "NIST 2017 language recognition evaluation plan," 2017, available at https://www.nist.gov/system/files/documents/2017/09/29/lre17_eval_plan-2017-09-29_v1.pdf.

[12] J. Valk and T. Alumäe, "Voxlingua107: A dataset for spoken language recognition," in *2021 IEEE Spoken Language Technology Workshop (SLT)*, 2021, pp. 652–658.

[13] L. N. Smith, "Cyclical learning rates for training neural networks," *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 464–472, 2015.

[14] J. Thienpondt, B. Desplanques, and K. Demuynck, "Integrating frequency translational invariance in tdnns and frequency positional information in 2d resnets to enhance speaker verification," in *Interspeech*, 2021.

[15] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5796–5800, 2019.

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

[17] J. Villalba, N. Chen, D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, B. Borgstrom, F. Richardson, S. Shon, F. Grondin, R. Dehak, P. Garcia, D. Povey, P. Torres-Carrasquillo, S. Khudanpur, and N. Dehak, "State-of-the-art speaker recognition for telephone and video speech: The jhu-mit submission for nist sre18," in *Proceedings of Interspeech*, 2019, pp. 1488–1492.

[18] B. Desplanques, J. Thienpondt, and K. Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," in *Interspeech*, 2020.

[19] J. Deng, J. Guo, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4685–4694, 2019.

[20] X. Xiang, S. Wang, H. Huang, Y. Qian, and K. Yu, "Margin matters: Towards more discriminative deep neural network embeddings for speaker recognition," *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 1652–1656, 2019.

[21] 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.

[22] Y. Zhang, Z. Lv, H. Wu, S. Zhang, P. Hu, Z. Wu, H. yi Lee, and H. M. Meng, "Mfa-conformer: Multi-scale feature aggregation conformer for automatic speaker verification," in *Interspeech*, 2022.

[23] H. Wang, Y. Wang, Z. Zhou, X. Ji, Z. Li, D. Gong, J. Zhou, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5265–5274, 2018.

[24] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," *arXiv preprint arXiv: 1410.7455*, 2015.

[25] D. Garcia-Romero, G. Sell, and A. McCree, "Magneto: X-vector magnitude estimation network plus offset for improved speaker recognition," in *Odyssey*, 2020.