FINE-TUNING SELF-SUPERVISED MODELS FOR LANGUAGE IDENTIFICATION USING ORTHONORMAL CONSTRAINT

Amrutha Prasad[1,2], *Andrés Carofilis*[3], *Geoffroy Vanderreydt*[4], *Driss Khalil*[1], *Srikanth Madikeri*[1], *Petr Motlicek*[1,2], *Christof Schuepbach*[3]

[1] Idiap Research Institute, Martigny, Switzerland [2] Brno University of Technology, Brno, Czech Republic
[3]University of León, León, Spain [4]IDLab, Ghent University - imec, Ghent, Belgium
[5] Armasuisse Science and Technology, Thun, Switzerland

ABSTRACT

Self-supervised models trained with high linguistic diversity, such as the XLS-R model, can be effectively fine-tuned for the language recognition task. Typically, a back-end classifier followed by statistics pooling layer are added during training. Commonly used back-end classifiers require a large number of parameters to be trained, which is not ideal in limited data conditions. In this work, we explore smaller parameter back-ends using factorized Time Delay Neural Network (TDNN-F). The TDNN-F architecture is also integrated into Emphasized Channel Attention, Propagation and Aggregation-TDNN (ECAPA-TDNN) models, termed ECAPA-TDNN-F, reducing the number of parameters by 30 to 50% absolute, with competitive accuracies and no change in minimum cost. The results show that the ECAPA-TDNN-F can be extended to tasks where ECAPA-TDNN is suitable. We also test the effectiveness of a linear classifier and a variant, the Orthonormal linear classifier, previously used in x-vector type systems. The models are trained with NIST LRE17 data and evaluated on NIST LRE17, LRE22 and the ATCO2 LID datasets. Both linear classifiers outperform conventional back-ends with improvements in accuracy between 0.9% and 9.1%.

Index Terms— Language Identification, Transformers, Wav2Vec2, fine-tuning, low-resource, out-of-domain,

1. INTRODUCTION

Language Identification (LID) aims to automatically identify the language spoken in a given speech segment. While numerous studies have achieved impressive results in LID task by using large, well-balanced datasets, the task remains challenging when dealing with low-resource and out-of-distribution datasets. In such scenarios, the scarcity of labeled examples and the presence of linguistic variations pose significant hurdles in developing practical LID systems.

This work was supported by Armasuisse Science and Technology.

Fine-tuning large, self-supervised, pre-trained models, such as wav2vec 2.0 [1], has been shown to significantly improve performances on many downstream tasks [2, 3, 4] including LID [5, 6]. When fine-tuning such models for LID, a classification module followed by a statistics pooling module [7] are added prior to the final softmax layer. Common choices for model architectures used for the classification are: linear classifiers [2], transformers [8], TDNN and Emphasized Channel Attention, Propagation and Aggregation in Time Delay Neural Network (ECAPA-TDNN) [9]. Transformers and ECAPA-TDNN have been shown to provide superior performance over linear classifiers when there is sufficient data to train. However, classifiers with fewer parameters may be preferred in conditions with limited number of classes and data to fine-tune, a typical case in language recognition evaluations [10, 11].

Given the effectiveness of TDNN-based classifiers, this paper explores the use of factorized TDNN layers (TDNN-F, [12]) for LID, which have been shown to effectively combine the contextual modelling capacity of TDNN, while significantly limiting the size of the models. TDNN-F models that follow wav2vec 2.0 models, such as the XLSR-53, have been effectively used for speech recognition [3].

In this paper, we propose to apply Factorized TDNN models for LID, resulting in two classification modules: (1) TDNN-F, and (2) ECAPA-TDNN-F, where TDNN-F replaces the TDNN components in ECAPA-TDNN achieving parameter reduction up to 50%. In addition, we propose to exploit the orthonormal constraint employed in the TDNN-F architecture for the linear classifier mentioned earlier. The constraint alleviates the strong diagonal covariance assumptions in the statistics pooling layer, demonstrated with x-vector based systems [13].

We compare the performance of the XLS-R model [14] on low-resource and out-of-domain data conditions for LID. The model is fine-tuned with the LRE17 dataset, and evaluated on two unseen and out-of-distribution sets – LRE22 and air-traffic

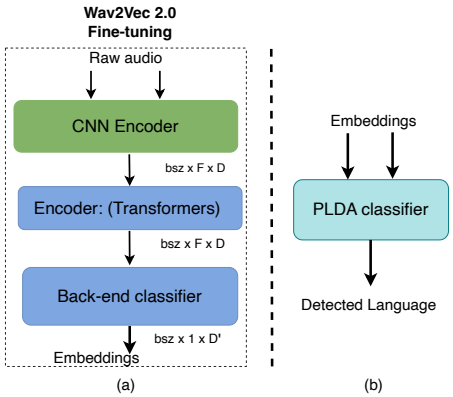

Fig. 1. Overview of the fine-tuning pipeline for LID. The back-end classifier block represents the different architectures explored in this work. The two stages of LID: (a) embeddings generation and, (b) language classification.

communication data [1](collected using very high frequency receives and thus very noisy. Our experiments show that the Orthonormal linear classifier performs the best on all test sets.

Section 2 describes the baseline approaches for fine-tuning pre-trained models for LID. In Section 3, we described the proposed approaches that apply TDNN-F and Orthonormal constraints to the back-end classifiers. Section 4 describes the training, development and test sets used in this work. The results are presented in Section 5. An overview of the findings in this work is provided in Section 6.

2. FINE-TUNING XLS-R FOR LID

Prior work on LID has been significantly shaped by innovative technologies from speaker recognition tasks, such as use of SGMMs [15], i-vectors [16], online i-vectors [17] dealing with short segments, or phonetically aware d-vectors [18]. In our experiments we fine-tune the pre-trained 300M parameter version of the XLS-R [14] model[2] that has been pre-trained with 128 languages and around 430,000 h of data using the espresso [19] toolkit which internally uses fairseq [20]. As shown in Figure 1, the XLS-R model is employed as an encoder and fine-tuned with LRE17 train data to extract embeddings. Further, a Probabilistic Linear Discriminant Analysis (PLDA) [21] classifier is trained with the dev set of each dataset (depending on the evaluation condition) using the embeddings generated from the fine-tuned model. As a part of our baseline systems, we use the following classifier architectures: linear, ECAPA-TDNN, Transformer. Each of these approaches are described below:

[1] https://atco2.org/

[2] https://github.com/facebookresearch/fairseq/tree/main/examples/wav2vec/xlsr

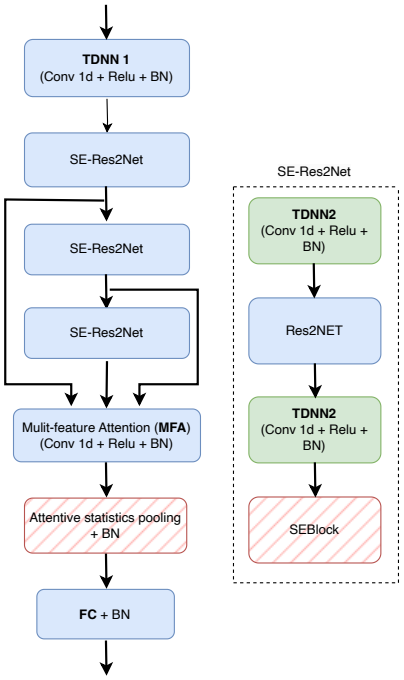

Fig. 2. Overview of the ECAPA-TDNN model architecture. Different colours indicate the possibility of applying the orthonormal constraint in the ECAPA-TDNN-F model for each block. Green: orthonormal constraint is always applied. Blue: orthonormal constraint is applied in certain configurations. Shaded Red: the constraint is not applied.

Linear layer: A linear layer of output dimension 512 is added prior to statistics pooling.

ECAPA-TDNN [9]: In speaker verification and language identification tasks, ECAPA-TDNN model have shown to provide state-of-the-art performance. We use the SpeechBrain [22] implementation along with their default configuration.

Transformer Layer [8]: a single transformer layer with the same configuration as other transformer layers in the XLS-R model is added after the XLS-R encoder.

In order to generate the audio-level embeddings, we employ statistics pooling mechanism for all the components mentioned above. The embeddings generated from the fine-tuned models is then used for the PLDA classifier and scored using the Kaldi [23] toolkit.

3. ORTHONORMALLY CONSTRAINED CLASSIFIERS

The orthonormal constraint for TDNN was introduced in [12]. The resulting TDNN model, namely factorized TDNN (TDNN-F), has been successfully applied for many speech applications:

speech recognition, speaker and language recognition. Each TDNN-F layer reduces the number of parameters in the TDNN module by applying the orthonormal constraint (inspired by Singular Value Decomposition). In this technique, the weight parameters $\mathbf{W}$, say of dimension $D_1 \times D_2$, of the linear layer are split into $\mathbf{BA}$ where, $\mathbf{B} \sim D_1 \times d$ and $\mathbf{A} \sim d \times D_2$. Orthonormal constraint is applied on $\mathbf{A}$ during training such that $\mathbf{A}^T\mathbf{A} \approx \mathbf{I}$. It is easy to see that the parameter reduction can be obtained by setting a low value for d. In the Kaldi implementation, the value for d is typically 160. We used the Pytorch implementation of this constraint from Pkwrap [24], which applies the constraint in every training iteration (in the original implementation in Kaldi parameters are updated with probability 0.25). The learning rate is updated accordingly.

We propose to use three classifiers based on the orthonormal constraint:

Orthonormal Linear layer: The linear layer of an output dimension 512 is subject to orthonormal constraint. Note that there is no parameter reduction in this case. The orthonormal transform is only intended to compute diagonal covariance in the statistics pooling layer. This operation was typical in x-vector based systems [23].

ECAPA-TDNN-F: Figure 2 shows the ECAPA-TDNN model architecture. We apply orthonormal constraint to the Conv1d layers of the model. Since we use the implementation from Speechbrain (including hyper-parameter settings), we override the Conv1d to recast the weights from $D_1 \times D_2 \times k$ to first $D_1 \times k \times D_2$ and then to $D_1 \times kD_2$. The orthonormal constraint is then applied on this weight matrix. Here, D_1 is the number of out channels, D_2 is the number of in channels and k is the kernel size. We experiment with the application of the parameter reduction in different components in the model architecture.

TDNN-F [12]: We also experiment with the TDNN-F architecture following its use in ASR with the XLSR LF-MMI architecture [3]. A context size of 3 is used for all the layers, with bottleneck dimension 160 and layer dimension 1024.

4. DATASETS

This sections briefly describes the datasets used for training and evaluation of the LID systems.

LRE17: The NIST Language Recognition Evaluation dataset consists of 14 languages and 3 parts: train, dev, and eval [10]. The splits contain 2061 h, 21 h, and 236 h of data respectively. Three-way speed perturbation [25] is applied on the LRE17 train data for fine-tuning. We filter out the LRE17 dev data to use only the audio files that are less than 100 s. On the LRE17 eval data, silence removal is applied on all audios that have length more than 100 s.

LRE22 [11]: The data comprises of 14 languages from African countries. The dev and test splits contain 30 h and 193 h of audio respectively. The evaluation focused on developing technologies to improve LID for low-resource languages given the average amount of dev data for each language is around 2 h.

ATCO2: The ATCO2 LID data was collected as a part of the ATCO2 project to develop and evaluate techniques for English/Non English – being Czech and French – classification [26]. This set is considered as the out-of-distribution data as it belongs to a completely different domain compared to LRE17 and LRE22, and the amount of annotated data is limited. In addition, the speech data obtained from the air-traffic communication domain can be extremely noisy. The ATCO2 data is annotated for LID and is split between development (14.4 h) and evaluation sets (11.6 h). ATCO2 is publicly available [3]. In our experiments, Czech, English and French languages are used for evaluations.

For each eval set, the corresponding dev data is used to train the PLDA classifier.

5. EXPERIMENTS

In all our experiments, we fine-tune the XLS-R model with the LRE17 train split with a learning rate of 3e-05 for 42'000 steps with a batch size of 8 along with the cross-entropy loss. The back-end classifiers are trained with a gradient multiplier 20. We report the LID Accuracy(%), F1-score for our experiments. We also report min.C for LRE17 and LRE22 which is a metric provided by NIST [11].

Table 1 presents the results of using different back-end classifier architectures. The results show that systems using only linear layers (Linear layer in baseline, Orthonormal linear in proposed methods) consistently outperforms other baseline back-end classifier architectures. This could be attributed to the low-resource nature of the fine-tuning set. Moreover, the linear layers generalize better to out of domain sets. The Orthonormal linear performs better in two out of three conditions compared to the linear layer, with absolute improvement in accuracy up to 0.9% (relative improvement in error of 9%) and 1% in F-1 score. Although ECAPA-TDNN and transformer back-ends provide competitive results, the number of parameters during fine-tuning increases significantly (from 8 to 10 times).

The TDNN-F model outperforms the ECAPA-TDNN and Transformer models on only the seen condition set with improvements in accuracy up to 2.6% (13.2% relative improvement in error rate), indicating its tendency to overfit. We also compared the LRE17 with the baseline x-vector system pre-

[3] https://www.atco2.org/data

Table 1. Results of LRE17, LRE22 and ATCO2 test sets evaluated with different back-end configurations. The number of parameters is measured only on the back-end classifiers. All systems are trained on the LRE17 train set, but dataset-specific PLDA models are trained for LID. min.C: minimum decision cost function for LRE datasets. *: for the configuration of ECAPA-TDNN-F models please refer to Table 2.

	Back-end	No. of	**Accuracy(%)**			**F1-score**			**min.C**	
	Config	params (M)	LRE17	LRE22	ATCO2	LRE17	LRE22	ATCO2	LRE17	LRE22
Baselines	Linear layers	0.5	83.1	**70.4**	90.1	0.83	**0.72**	0.79	0.20	**0.30**
	ECAPA-TDNN	10.0	80.4	59.7	87.7	0.81	0.62	0.73	0.22	0.42
	Transformer layer	9.0	81.8	61.3	90.5	0.81	0.63	0.79	0.22	0.40
Proposed	Orthonormal linear	0.5	**83.7**	67.0	**91.0**	**0.84**	0.68	**0.80**	**0.19**	0.35
	TDNN-F	2.0	83.0	56.8	85.9	0.83	0.59	0.70	**0.19**	0.45
	ECAPA-TDNN-F*	7.0	81.7	58.8	87.4	0.82	0.62	0.74	0.20	0.44
	ECAPA-TDNN-F*	5.0	80.8	57.9	87.2	0.81	0.6	0.73	0.22	0.44
Other baselines	x-bLSTM [27]	-	68.7	-	-	-	-	-	-	-
	Whisper (medium)	-	-	-	56.4	-	-	-	-	-

Table 2. Results of LRE17, LRE22 and ATCO2 test sets evaluated with different bottleneck configuration for ECAPA-TDNN-F. A bottleneck dimension of 16 is used for Res2Net and 128 for the other blocks. See Figure 2 for the module details.

Constrained	**No. of**	**Accuracy(%)**		
module	**params (M)**	LRE17	LRE22	ATCO2
None	10.0	80.4	59.7	87.7
TDNN2	9.3	80.8	58.8	87.6
+ Res2Net	9.0	80.0	58.7	87.7
+ TDNN1	7.0	81.7	58.8	87.4
+ MFA	5.0	80.8	57.9	87.2
+ FC	4.0	77.5	56.1	63.0

sented in [27] and observed the increase in the accuracy by 9.2% absolute. On the ATCO2 set, the accuracy was 91% compared to the Whisper model [28] which provides an accuracy of 56.4%.

Next, we compare the results of ECAPA-TDNN-F and ECAPA-TDNN in Table 1. The results show that the performance of ECAPA-TDNN can be preserved even after reducing the parameters by 30%, and when reduced drastically by 50% we observe a relative degradation in accuracy by 3% for LRE22 and by 0.5% for ATCO2. However, no degradation in min cost was observed. Unlike TDNN-F, the ECAPA-TDNN-F maintains its performance on unseen conditions compared to ECAPA-TDNN.

Table 2 shows the effect of applying TDNN-F layers to various TDNN modules in ECAPA-TDNN model. Specifically, we study the effect of replacing with TDNN-F components. Our results indicate that the replacement is effective (i.e., reduces parameters without significantly affecting performance), as long as the final FC layer is not parameter deficient. Our results opens up new avenues of exploring the use of ECAPA-TDNN-F in tasks where ECAPA-TDNN has been shown to be effective. The severe degradation for ATCO2 when using TDNN-F for the FC component can be attributed to the size (10x smaller than LRE22) and noisy nature of the dataset.

6. CONCLUSIONS

In this paper, we explored different back-end classifier architecture for XLS-R based LID: Linear, ECAPA-TDNN, Transformer, Orthonormal Linear, TDNN-F and ECAPA-TDNN-F. The models were trained with LRE17 train data and evaluated on LRE17 test set and two other low-resource (LRE22) and out-of-distribution (ATCO2) datasets. With the Orthonormal linear classifier, by adding a simple linear layer with an orthonormal constraint we obtained improvements in accuracy and F1 score in two out of three conditions over using a linear layer in the back-end. When fine-tuning with limited data, a linear layer – with or without Orthonormal constraint – outperforms other common architecture choices. Using only TDNN-F instead of ECAPA-TDNN or the Transformer improved the in-domain performance by up to 2.3% absolute. Finally, ECAPA-TDNN-F replaces the TDNN layers in the ECAPA-TDNN with TDNN-F layers reducing number of parameters by up to 50%. When reducing by only 30% no significant degradation was observed compared to ECAPA-TDNN, and with 50% reduction we still achieved competitive results.

7. REFERENCES

[1] Alexei Baevski et al., "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Ad-*

vances in neural information processing systems, vol. 33, pp. 12449–12460, 2020.

[2] Alexis Conneau et al., "Unsupervised cross-lingual representation learning for speech recognition," *arXiv preprint arXiv:2006.13979*, 2020.

[3] Apoorv Vyas, Srikanth Madikeri, and Hervé Bourlard, "Lattice-free mmi adaptation of self-supervised pretrained acoustic models," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6219–6223.

[4] Zhiyun Fan, Meng Li, Shiyu Zhou, and Bo Xu, "Exploring wav2vec 2.0 on speaker verification and language identification," *arXiv preprint arXiv:2012.06185*, 2020.

[5] Andros Tjandra et al., "Improved language identification through cross-lingual self-supervised learning," in *Proc. of ICASSP*. IEEE, 2022, pp. 6877–6881.

[6] Jiatong Shi et al., "Ml-superb: Multilingual speech universal performance benchmark," *arXiv preprint arXiv:2305.10615*, 2023.

[7] David Snyder et al., "Spoken language recognition using x-vectors.," in *Odyssey*, 2018, vol. 2018, pp. 105–111.

[8] Ashish Vaswani et al., "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[9] Brecht Desplanques et al., "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," *arXiv preprint arXiv:2005.07143*, 2020.

[10] Seyed Omid Sadjadi et al., "The 2017 NIST language recognition evaluation.," in *Odyssey*, 2018, pp. 82–89.

[11] Yooyoung Lee et al., "NIST 2022 language recognition evaluation plan," 2022.

[12] Daniel Povey et al., "Semi-orthogonal low-rank matrix factorization for deep neural networks.," in *Proc. of Interspeech*, 2018, pp. 3743–3747.

[13] David Snyder et al., "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.

[14] Arun Babu et al., "XLS-R: Self-supervised cross-lingual speech representation learning at scale," in *Proc. of Interspeech*, 2022, pp. 2278–2282.

[15] Petr Motlicek et al., "Employment of subspace gaussian mixture models in speaker recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4445–4449.

[16] Najim Dehak et al., "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[17] Subhadeep Dey et al., "Template-matching for text-dependent speaker verification," *Speech communication*, vol. 88, pp. 96–105, 2017.

[18] Subhadeep Dey et al., "Dnn based speaker embedding using content information for text-dependent speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5344–5348.

[19] Yiming Wang et al., "Espresso: A fast end-to-end neural speech recognition toolkit," in *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 136–143.

[20] Myle Ott et al., "fairseq: A fast, extensible toolkit for sequence modeling," *arXiv preprint arXiv:1904.01038*, 2019.

[21] Simon JD Prince and James H Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *2007 IEEE 11th international conference on computer vision*. IEEE, 2007, pp. 1–8.

[22] Mirco Ravanelli et al., "SpeechBrain: A general-purpose speech toolkit," 2021, arXiv:2106.04624.

[23] Daniel Povey et al., "The Kaldi speech recognition toolkit," in *Proc. of IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011, number CONF.

[24] Srikanth Madikeri et al., "Pkwrap: a pytorch package for lf-mmi training of acoustic models," *arXiv preprint arXiv:2010.03466*, 2020.

[25] Tom Ko et al., "Audio augmentation for speech recognition," in *Sixteenth annual conference of the international speech communication association*, 2015.

[26] Igor Szoke et al., "Detecting english speech in the air traffic control voice communication," *arXiv preprint arXiv:2104.02332*, 2021.

[27] Bharat Padi et al., "Towards relevance and sequence modeling in language recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1223–1232, 2020.

[28] Alec Radford et al., "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28492–28518.