INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Language Recognition using Triplet Neural Networks

Victoria Mingote[1,2], *Diego Castan*[2], *Mitchell McLaren*[2], *Mahesh Kumar Nandwana*[2], *Alfonso Ortega*[1], *Eduardo Lleida*[1], *Antonio Miguel*[1]

[1]ViVoLab, Aragón Institute for Engineering Research (I3A), University of Zaragoza, Spain
[2]Speech Technology and Research Laboratory, SRI International, California, USA

{vmingote,ortega,lleida,amiguel}@unizar.es,
{diego.castan,mitchell.mclaren,mahesh.nandwana}@sri.com

Abstract

In this paper, we propose a novel neural network back-end approach based on triplets for the language recognition task, due to its success application in the related field of text-dependent speaker verification. A triplet is a training example constructed of three audio samples; two from the same class and one from a different class. In presenting two pairs of samples to the network, the triplet neural network learns to discriminate between samples from the same languages and pairs of different languages. Triplet-based training optimizes the Area Under the Curve (AUC) in contrast to other triplet loss functions proposed in the literature. The optimization of the AUC as cost function is appropriate for a detection task as it directly correlates with end-use performance of the system. Moreover, we show the importance of defining an appropriate method of triplet selection and how this impacts performance of the system. When benchmarked on the LRE09 database, the new triplet back-end demonstrated superior performance compared to traditional back-ends used for language recognition. In addition, we performed an evaluation on the LRE15 and LRE17 databases to check the generalization power of the proposed systems.

Index Terms: Language Recognition, Triplet Neural Network, Hard Negative Mining, PLDA, SVM, GB

1. Introduction

Spoken language recognition involves correctly identifying the spoken language of an audio file. Due to similarity in the research fields, recent progress in automatic speech recognition and speaker recognition techniques based on discriminative deep neural networks (DNN) [1, 2, 3, 4, 5] have subsequently improved the technology applied to language recognition. Existing systems in the literature achieve good performance when the languages are easily separable, but struggle to separate closely related languages (i.e., dialects) [6]. Thus, we propose to attend to this issue using one of the most recent approaches developed in deep learning face and speaker recognition systems [7, 8, 9, 10] — a triplet neural network. For training, triplet neural networks define a loss function that aims to maximize the similarity between a pair of examples belonging to the same identity or language, while minimizing the similarity with an example from another subject or language. Following the philosophy of this kind of network and applying the new loss function that we proposed in [11], we will show that the training of a system with this approach is more suitable than traditional back-ends for the language recognition task.

For many years, language recognition systems have been based on i-vectors [12, 13]. In this approach, the system pipeline consists of two main blocks; a front-end formed by a Gaussian Mixture Model (GMM) as a Universal Background Model (UBM) along with a total variability subspace for i-vector extraction, and a back-end, which may be one of the diverse classification techniques to produce system scores. More recently, modern systems have substituted parts of the previous pipeline with DNNs to achieve improved robustness to varying conditions and audio duration. For instance, a significant step forward in language recognition was obtained by replacing acoustic features with DNN bottleneck representations or replacing UBM posteriors with DNN senone posteriors [14, 15] within the i-vector framework. Other more ambitious DNN approaches have been used including end-to-end systems [16, 17]. These techniques consist of a DNN trained with a softmax layer at the output and a multiclass cross-entropy objective, so the network learns to directly classify the languages.

These end-to-end DNN systems are rather successful in this task [16, 17], but the approach requires relatively large training datasets compared to the i-vector framework, and it is computationally more expensive. More recently, leveraging advances in speaker recognition, the use of DNN language embeddings to produce an utterance-level representation of the speech has become commonplace [18]. The embeddings replace the i-vectors as front-end [19, 20, 21]. In this approach, once the DNN is trained, the embeddings representations are obtained from one of the previous layers inside of the DNN, and these are evaluated using a back-end classifier such as a Gaussian Back-end (GB) [15], Logistic Regression [12], Support Vector Machines (SVM) [22], Probabilistic Linear Discriminant Analysis (PLDA) [23], or Neural Networks (NN) [24].

In this paper, we propose to apply a new approach in the language recognition task that combines the triplet loss with the Area Under the Curve (AUC) function to improve the discrimination ability between similar languages. We focus our attention on improving the back-end part of the system, due to the assumption that the language embedding front-end can generalize to a variety of domains. Besides, working only in the back-end allows us to adapt faster the system for specific needs. To demonstrate the advantages of the triplet neural network back-end, we benchmark against traditional back-ends included GB, PLDA, SVM, and NN.

This paper is laid out as follows. Section 2 provides a description of the system, including traditional and back-ends used in this study. The proposed triplet back-end approaches are details in Section 3, with the experiment protocol in Section 4. Results and analysis are given in Section 5. Conclusions are presented in Section 6.

2. System Description

The following section describes the system used in this work. We first detail the front-end used to extract a language embedding from an audio sample. This is followed by a description of

http://dx.doi.org/10.21437/Interspeech.2019-2437

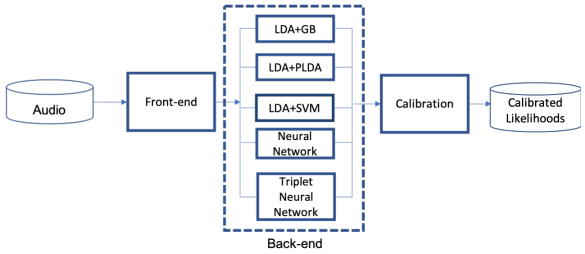

Figure 1: *The language recognition system, composed of a front-end, selection of one back-end option, and calibration.*

various back-ends explored in this work, with each being based on the same embeddings extracted from the front-end. A final calibration step is applied to scores from each back-end in order to transform the scores into calibrated likelihoods.

2.1. Front-end

The front-end is composed of 3 subsystems. The Speech Activity Detection (SAD) system selects the frames that contain speech, then a Bottleneck Senone DNN (BN-DNN) system extracts features that are rich in phonetic information. The BN features are used as input in an embeddings DNN trained to discriminate languages as the output classes. The final embeddings used by the back-end are extracted from this DNN.

2.1.1. Speech Activity Detection (SAD)

We use a DNN-SAD model composed of two hidden layers with 500 and 100 nodes, respectively, and trained to discriminate speech vs non-speech frames. It is based on 20-dimensional MFCC features, stacked with 31 frames to provide a 620-dimensional input to the DNN, which is first mean and variance normalized over a 201 frame window. A threshold of 0.0 was applied to speech posteriors to determine if a frame of audio consisted of speech or not.

2.1.2. Senone DNN Bottleneck Extractor

The senone DNN extracts 80-dimensional bottleneck (BN) features from a bottleneck layer. The network targets 3450 senones (tri-phone states) learned using Fisher and Switchboard data. Log Mel Spectra features of 40 dimensions were used as acoustic input features. The network has 5 hidden layers of 600 nodes with the last hidden layer being the bottleneck of 80 nodes.

2.1.3. Language embeddings extractor

The embeddings network is trained on BN features extracted from the BN-DNN and is trained to discriminate 49 different languages from the LRE09 development data. The data was augmented with 4 types of degradation; reverberation, compression, non-vocal music, and noise. The structure of the network starts with four frame-level hidden layers followed by a statistics pooling layer and two final segment layers. The features for the back-end are extracted from the first segment-level hidden layer of 512 nodes. More information about the DNN structure and augmentation process can be found in [25].

2.2. Traditional Back-ends

To establish a baseline framework, we applied a range of traditional language recognition back-ends to the language embeddings. Before applying each of these techniques, embeddings were transformed to 48 dimensions with Linear Discriminant Analysis (LDA) into a new feature space that maximizes class separability, also followed by mean and length normalization.

2.2.1. Weighted Gaussian Back-end (WGB)

The Generative Gaussian Back-end is one of the most common back-ends applied to language recognition. In this back-end, each language is modeled by a Gaussian distribution, defined by a language-dependent mean and a full covariance matrix shared across all languages. These Gaussian models are used to compute the likelihood of an embedding being based on each of the modeled languages. Since the back-end training data is language-imbalanced, we use a modification of the Gaussian Back-end [15] to normalize for the imbalance by appropriately weighting the examples during the computation of the means m_l and covariance S of the model. The weighted GB is thus defined by,

$$m_l = \frac{\sum_{i|l_i=l} w_i \cdot e_i}{\sum_{i|l_i=l} w_i} \tag{1}$$

$$S = \frac{\sum_l \sum_{i|l_i=l} w_i \cdot (e_i - m_l)^T \cdot (e_i - m_l)}{\sum_i w_i}, \tag{2}$$

where e_i is the embedding vector with $i \in \{1, ..., n\}$ and n is the total number of examples, w_i is the weight assigned to each example, and l_i is the language present in the example. The weights are computed such that all samples from a language are weighted equally ($w_i = w_j$ if $l_i = l_j$) and the sum of their weights is identical across languages ($\sum_{i|l_i=l} w_i$ is the same for all l).

2.2.2. Probabilistic Linear Discriminant Analysis (PLDA)

Traditionally, PLDA approaches combined with i-vectors were not as successful for language recognition as in speaker recognition tasks, since the number of languages *N* is much smaller than the speaker identities so there is a significant loss of information for the identification process with the projection into a *(N-1)* dimensional space. In view of the impressive results achieved with the PLDA in speaker verification tasks with embeddings, we apply it to have a reference result of PLDA in embeddings for language recognition. Thus, we use as a binary detector to determine whether pairs of examples are from the same language or different. The full PLDA modeling that we employ can be expressed as

$$y_i = \mu + U_1 \cdot x_1 + U_2 \cdot x_{2i} + \epsilon_i, \tag{3}$$

where μ is the language-independent mean vector, U_1 is the eigenlanguage matrix, x_1 is the language factor, U_2 is the eigenchannel matrix, x_{2i} is the channel factor with $i \in \{1, ..., n\}$ and n is the total number of examples, and ϵ_i models the residual variability.

2.2.3. Support Vector Machines (SVM)

The aforementioned generative back-end techniques may struggle to separate similar languages because they are not trained to explicitly separate them. For this reason, we decided to substitute the classical generative back-end with discriminative techniques. Before starting with the neural network approaches, we employ a discriminative SVM kernel based classifier [22]. The SVM classifier focuses on maximizing the margin between language classes. This classifier is constructed from the sum of a kernel function as

$$f(e) = \sum_i \alpha_i \cdot y_i \cdot K(e, e_i) + b, \tag{4}$$

where e_i is the embedding vector with $i \in \{1, ..., n\}$ and n is the total number of examples, α_i are the learned coefficients, y_i are the target values, and $K(e, e_i)$ is the kernel function used

to discriminate languages. In our case, the kernel function employed was radial basis function (rbf).

The SVM is by nature a binary classifier, so in order to use the SVM with multiclass data we employ a "one vs. all" strategy. This consists of training language-dependent SVMs that target one language against the pool of all other languages. In this manner, the margin between the target language and other languages will be largely defined by closely-related languages to the target language.

2.2.4. Neural Network Back-end

We also apply a feed-forward neural network approach, which is trained to discriminate languages directly. In contrast to the embeddings network trained on short audio cuts of 2-4 seconds, this back-end is trained using longer audio samples using a fixed embeddings extraction process. The NN employed is trained using a cross-entropy cost function for multi-class language classification. After the training process, a secondary embedding is extracted from an intermediate layer and a similarity metric such as cosine similarity is applied to compare the test embedding to those obtained from the embeddings of each language in the training dataset.

2.3. Calibration

The scores generated by all the back-ends are calibrated by transforming the scores into proper likelihoods using multiclass logistic regression as described in [26]. These are then transformed into calibrated log-likelihood ratios (LLRs).

3. Triplet Neural Network Back-end

Previous work found that language-imbalanced databases are not well suited to neural network back-end techniques, which is a hinderance when a large amount of training data for each language is required [24, 15]. Thus, to help with this imbalance, we propose to apply a novel approach for the language recognition task based on a triplet neural network that is trained to discriminate between pairs of embeddings.

The triplet neural network consists of the same neural network replicated three times with shared parameters, which are then used to evaluate with a cost function the embeddings provided by these three instances. In our case, the input of this network are three embeddings, an embedding from a specific language e (an anchor), another embedding from the same language e^+ (a positive example), and an embedding from another language e^- (a negative example). For training the triplet neural network back-end, the proposed scheme is depicted in Fig.2.

Traditionally, in most of the existing speaker recognition systems using this approach [9, 10, 27, 28], the goal of the neural network learning process is to maximize the distance between the anchor and the negative example while the distance between the anchor and the positive example is minimized, if it is greater than a margin defined in the triplet loss function.

Unlike the previous systems, we showed in [11] that for the recognition task, direct optimization of the AUC metric is a better and more intuitive option than traditional metrics, since this metric is a measure of the system performance. Thus, we have also used this approach for the language recognition task with the differentiable approximation of the AUC function applied in [11] to find the network parameters θ. This maximizes the function,

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} \frac{1}{m^+ m^-} \sum_{i}^{m^+} \sum_{j}^{m^-} \sigma(\alpha(s_\Theta(p_i^+) - s_\Theta(p_j^-))), \tag{5}$$

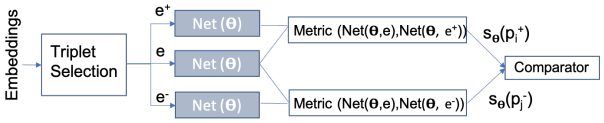

Figure 2: *Triplet neural network where the embeddings are grouped in triplets to train the network and evaluated the two pairs of embeddings to optimize the objective function.*

where $\sigma(s)$ is the sigmoid function, $s_\Theta(p_i^+)$ is the similarity metric of each pair of anchor-positive embeddings where $p_i^+ = (e, e_i^+)$ with $i \in \{1, ..., m^+\}$ and m^+ is the total number of positive examples, $s_\Theta(p_j^-)$ indicates the metric of each pair of anchor-negative embeddings where $p_j^- = (e, e_j^-)$ with $i \in \{1, ..., m^-\}$ and m^- is the total number of negative examples, and α is an adjustable parameter.

Aside from the loss function, the method used to select triplets for training is an important factor in training a good triplet network. If the anchor-negative example is consistently from easily-separable languages, the network will not provide the discrimination needed to separate closely-related languages. We use two triplet selection strategies as described below.

3.1. Random Selection

The first one is a straightforward approach which consists of a random choice of the languages and the examples of each language. To perform this selection, in each iteration, we randomly choose G groups of P languages where $G \cdot P$ is equal to the total number of languages N. Then, all the examples from each selected language of each group are used to make random triplets to train the neural network.

3.2. Hard Negative Mining Selection

This is a similar approach to the triplet sampling strategy proposed in [8] termed hard negative mining. This technique is an intelligent selection process which involves selecting the anchor-negative pairs with the maximum similarity value (hard negative) for which the system triggers a false alarm, and the anchor-positive pairs with the minimum similarity value (hard positive) which the system can not detect (i.e., missed detection). To manage this selection process, we propose two methods using both the same random selection for the languages:

- *Hard Negative Mining Type 1*: The first method was developed to create the triplets only with a random subset of K examples from each P language.
- *Hard Negative Mining Type 2*: As an evolution of the first approximation, we employ the whole set of examples from each P language to find the cases which are more complicated to discriminate. The benefit of this process in terms of performance comes with a higher computational cost.

4. Experimental Data

In this work, the experiments have been performed with the database that was provided by NIST for the LRE 2009 [29]. Specifically, we use a division of LRE09 development data that was proposed in [30]. The 49 original LRE09 development languages were used for training the different back-ends. For testing, we used test segments from the 23 target languages each cut to consist of just 8 seconds of speech. We focus on the closed-set language recognition problem in which there are no out-of-set examples, since all test examples belong to one of the 23 target languages. Also, we used the evaluation part of the LRE15 [31] and the LRE17 [32] corpus to make a final test, since these

Table 1: *Comparison of different traditional back-end techniques on the LRE09 [29] eval data in terms of showing EER% and Cllr. Audio files contained 8 seconds of speech.*

Back-end	Cllr	EER%
WGB	0.229	3.48
PLDA	**0.135**	**3.07**
SVM	0.317	4.33

databases are composed by clusters of closely-related languages and one of the paper goals is the dialect discrimination.

5. Results

In this paper, we report the results of our experiments in terms of log-likelihood-ratio cost function (Cllr), which is an indicator of both calibration performance and the discriminative power of the system. As opposed to defining a single operating point and threshold for detections, Cllr summaries information from all operating points, with a lower value indicating better overall calibration and discrimination performance. We also report the equal error rate (EER) from pooled language trials, where a trial is a comparison of a test example to a single language detector.

5.1. Baseline

In this section, we compare the performance of the traditional back-end techniques described in Section 2, each of them with previous LDA reduction, mean and length normalization, to establish a baseline result. Table 1 details the EER and Cllr for the initial systems based on three different back-ends. We have found that although traditionally the i-vector and PLDA pipeline has not been the most suitable approach for the language recognition task, the embeddings followed by PLDA as binary detector is found to outperform the back-ends based on classifiers such as WGB and SVM. Thus, in the following section we use the PLDA back-end as baseline system.

5.2. Neural Network Approaches

In this section, we benchmark both the existing and proposed neural network approaches against the PLDA baseline. To develop these experiments, we implemented a straightforward architecture with only one dense layer for the two different neural network approaches, and we used cosine similarity to produce system scores for a test sample.

In the Table 2, we can see the results achieved on the LRE09 in terms of EER% and Cllr, where we compare the NN performance and the triplet neural network perfomance with the different strategies proposed in Section 2: random selection strategy (TripletNet-Rand), Hard Negative Mining Type 1 (TripletNet-HM1), and Hard Negative Mining Type 2 (TripletNet-HM2). While the traditional NN back-end is outperformed by the PLDA back-end, the TripletNet-Rand is comparable to PLDA, and the use of an intelligent selection strategy via hard negative mining brings reasonable improvements in Cllr. In the best case based on Cllr result, TripletNet-HM2, we achieve a relative improvement of 17% in terms of Cllr with respect to the PLDA baseline. While, in terms of EER%, the result of TripletNet-HM1 is better than TripletNet-HM2 with a relative improvement of 25% respect to the PLDA.

Furthermore, we also evaluated the PLDA baseline and the neural network approaches on the LRE15 and LRE17 to check the ability to generalize from our back-end approach. The results in Table 2 show that a relative improvement of 9% in terms of EER% is achieved on the LRE17 using the triplet neural network. Nevertheless, in the LRE15, the PLDA performance outperforms the triplet neural network results. The performance of the system with respect to the proposed triplet selection strategies shows a promising trend, since more elaborated selections are providing better results. In the case of LRE15, the french cluster is known to suffer from significant mismatch between development and evaluation sets [33, 34]. Therefore, we additionally present LRE15-nofre results in which the removal of the 'fre' cluster provided a 33% and 18% relative gain for the Triplet NN and PLDA backends, respectively. This demonstrates the triplet NN backend is particularly impacted by the condition mismatch in this cluster, and warrants further work to cope with this issue.

Table 2: *Comparison of PLDA and the triplet neural network approaches on the LRE09 [29], the LRE15 [31] and the LRE17 [32] eval data in terms of showing EER% and Cllr.*

Back-end	Results (Cllr / EER%)			
	LRE09	LRE15	LRE15-nofre	LRE17
PLDA	0.135/3.07	**0.231**/**5.91**	0.188/**4.60**	0.285/7.35
DNN	0.149/3.38	0.345/8.83	0.258/7.36	0.359/8.11
TripleNet − Rand	0.129/2.72	0.443/8.11	0.351/7.18	0.438/8.02
TripleNet − HM1	0.119/**2.29**	0.372/6.33	0.285/5.49	0.351/6.73
TripleNet − HM2	**0.112**/2.61	0.274/6.36	**0.183**/4.95	**0.283**/**6.72**

5.3. Limitations of the Triplet Neural Network Back-end

Despite the improvement clearly achieved in the system performance on the LRE09, we expect to obtain a relevant reduction of the confusion between closely related languages. We therefore conducted an analysis of the most confused pairs of languages based on the NIST LRE 2009 evaluation: Bosnian-Croatian, Farsi-Dari, and Russian-Ukrainian. Supporting our assumption, it was found that the confusion between Russian and Ukrainian languages has been reduced using the system with the best performance. Similarly, in the case of Farsi and Dari languages, Farsi test samples were more accurately allocated to Farsi instead of confused with Dari, however, the detection of Dari samples did not improve. Discrimination between Bosnian and Croatian did not tend to improve nor decline. A possible cause of the failure in the reduction of discrimination problem between these languages could be derived from the lack of examples of some of the languages mentioned. A possible solution to solve this problem would be to bias the triplet selection to provide relatively more triplets from the pairs of languages with greater confusion, with the intent to focus the discriminative power of the DNN on these languages.

6. Conclusions

In this paper, we have implemented a new triplet network approach to discriminate similar languages. This approach outperformed the traditional back-ends by up to 17% and 25% relative, in the context of a closed-set evaluation of the LRE 2009 dataset. Besides, this approach has provided relevant results in the LRE 2015 and 2017, although the generalization power can be still improved. Results confirm that a back-end technique based on triplet neural networks is an interesting line of research for this task since we have obtained competitive results even though some discrimination problems still exists, so we can achieve better results with an improvement of the language selection combined with the smart triplet selection.

7. Acknowledgements

This work has been supported by the Spanish Ministry of Economy and Competitiveness and the European Social Fund through the project TIN2017-85854-C4-1-R, and by the Government of Aragon (Reference Group T36_17R) and co-financed with Feder 2014-2020 "Building Europe from Aragon". This work has additionally been supported by SRI International internal research and development funding.

8. References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on audio, speech, and language processing*, vol. 20, no. 1, pp. 30–42, 2012.

[3] 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.

[4] G. Bhattacharya, M. J. Alam, and P. Kenny, "Deep Speaker Embeddings for Short-Duration Speaker Verification." in *Interspeech*, 2017, pp. 1517–1521.

[5] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep Neural Network Embeddings for Text-Independent Speaker Verification." in *Interspeech*, 2017, pp. 999–1003.

[6] M. Zampieri, S. Malmasi, N. Ljubei, P. Nakov, A. Ali, J. Tiedemann, Y. Scherrer, and N. Aepli, *Findings of the VarDial Evaluation Campaign 2017*, ser. Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects, 2017, iD: unige:94612. [Online]. Available: https://archive-ouverte.unige.ch/unige:94612

[7] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9370, no. 2010, pp. 84–92, 2015.

[8] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[9] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 165–170.

[10] C. Zhang and K. Koishida, "End-to-End Text-Independent Speaker Verification with Triplet Loss on Short Utterances." in *Interspeech*, 2017, pp. 1487–1491.

[11] V. Mingote, A. Miguel, A. Ortega, and E. Lleida, "Optimization of the Area Under the ROC Curve using Neural Network Supervectors for Text-Dependent Speaker Verification," *arXiv preprint arXiv:1901.11332*, 2019. [Online]. Available: http://arxiv.org/abs/1901.11332

[12] D. Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matějka, "Language recognition in ivectors space," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[13] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction," in *Twelfth annual conference of the international speech communication association*, 2011.

[14] R. Fér, P. Matějka, F. Grézl, O. Plchot, and J. Černockỳ, "Multilingual bottleneck features for language recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[15] L. Ferrer, Y. Lei, M. McLaren, and N. Scheffer, "Study of senone-based deep neural network approaches for spoken language recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 1, pp. 105–116, 2016.

[16] I. Lopez-Moreno, J. Gonzalez-Dominguez, O. Plchot, D. Martinez, J. Gonzalez-Rodriguez, and P. Moreno, "Automatic language identification using deep neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 5337–5341.

[17] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez-Rodriguez, and P. J. Moreno, "Automatic language identification using long short-term memory recurrent neural networks," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[18] S. O. Sadjadi, T. Kheyrkhah, A. Tong, C. Greenberg, D. Reynolds, E. Singer, L. Mason, and J. Hernandez-Cordero, "The 2017 NIST Language Recognition Evaluation ," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 82–89. [Online]. Available: http://dx.doi.org/10.21437/Odyssey.2018-12

[19] A. Lozano-Diez, O. Plchot, P. Matejka, and J. Gonzalez-Rodriguez, "Dnn based embeddings for language recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5184–5188.

[20] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken language recognition using x-vectors," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 105–111.

[21] J. Villalba, N. Brümmer, and N. Dehak, "End-to-end versus embedding neural networks for language recognition in mismatched conditions," in *Odyssey: The Speaker and Language Recognition Workshop, Les Sables dOlonne*, 2018.

[22] W. M. Campbell, E. Singer, P. A. Torres-Carrasquillo, and D. A. Reynolds, "Language recognition with support vector machines," in *ODYSSEY04-The Speaker and Language Recognition Workshop*, 2004.

[23] M. K. Rai, M. S. Fahad, J. Yadav, K. S. Rao *et al.*, "Language identification using PLDA based on I-vector in noisy environment," in *Advances in Computing, Communications and Informatics (ICACCI), 2016 International Conference on*. IEEE, 2016, pp. 1014–1020.

[24] M. McLaren, A. Lawson, Y. Lei, and N. Scheffer, "Adaptive Gaussian backend for robust language identification." in *INTERSPEECH*, 2013, pp. 84–88.

[25] M. McLaren, D. Castan, M. Nandwana, L. Ferrer, and E. Yilmaz, "How to train your speaker embeddings extractor," in *Odyssey: The Speaker and Language Recognition Workshop, Les Sables dOlonne*, 2018.

[26] D. A. Van Leeuwen and N. Brummer, "Channel-dependent gmm and multi-class logistic regression models for language recognition," in *2006 IEEE Odyssey-The Speaker and Language Recognition Workshop*. IEEE, 2006, pp. 1–8.

[27] S. Dey, S. Madikeri, and P. Motlicek, "End-to-end text-dependent speaker verification using novel distance measures," in *Proceedings Interspeech*, 2018, pp. 3598–3602.

[28] S. Novoselov, V. Shchemelinin, A. Shulipa, A. Kozlov, and I. Kremnev, "Triplet Loss Based Cosine Similarity Metric Learning for Text-independent Speaker Recognition," *Proc. Interspeech 2018*, pp. 2242–2246, 2018.

[29] "The 2009 NIST language recognition evaluation plan, 2009." [Online]. Available: http://www.itl.nist.gov/iad/mig/tests/lre/2009.

[30] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "Calibration Approaches for Language Detection." in *INTERSPEECH*, 2017, pp. 2804–2808.

[31] "The 2015 NIST language recognition evaluation plan, 2015." [Online]. Available: https://www.nist.gov/itl/iad/mig/upload/LRE15_Eval_Plan_v23.pdf.

[32] "NIST 2017 Language recognition evaluation plan, 2017." [Online]. Available: https://www.nist.gov/sites/default/files/documents/2017/09/29/lre17_eval_plan-2017-09-29_v1.pdf.

[33] K. Lee, H. Li, L. Deng, V. Hautamäki, W. Rao, X. Xiao, A. Larcher, H. Sun, T. Nguyen, G. Wang *et al.*, "The 2015 NIST language recognition evaluation: the shared view of I2R, Fantastic4 and SingaMS," in *Interspeech 2016*, 2016, pp. 3211–3215.

[34] O. Plchot, P. Matejka, R. Fér, O. Glembek, O. Novotnỳ, J. Pešán, K. Veselỳ, L. Ondel, M. Karafiát, F. Grézl *et al.*, "Bat system description for nist lre 2015," in *Proceedings of Odyssey*, 2016.