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

Combining Speaker Recognition and Metric Learning for Speaker-Dependent Representation Learning

João Monteiro[1,2], *Jahangir Alam*[1,2], *Tiago H. Falk*[1]

[1]Institut National de la Recherche Scientifique (INRS-EMT), Quebec, Canada
[2]Centre de Recherche Informatique de Montréal (CRIM), Quebec, Canada

joao.monteiro@emt.inrs.ca, jahangir.alam@crim.ca, falk@emt.inrs.ca

Abstract

In this paper, we tackle automatic speaker verification under a text-independent setting. Speaker modelling is performed by a deep convolutional neural network on top of time-frequency speech representations. Convolutions performed over the time dimension provide the means for the model to take both short-term dependencies into account, given the nature of the learned filters which operate over short-windows, as well as long-term dependencies, since depth in a convolutional stack implies dependency of outputs across large portions of input samples. Additionally, various pooling strategies across the time dimension are compared so as to effectively map varying length recordings into fixed dimensional representations while simultaneously providing the neural network with an extra mechanism to model long-term dependencies. We finally propose a training scheme under which well-known metric learning approaches, namely triplet loss minimization, is performed along with speaker recognition in a multi-class classification setting. Evaluation on well-known datasets and comparisons with state-of-the-art benchmarks show that the proposed setting is effective in yielding speaker-dependent representations, thus is well-suited for voice biometrics downstream tasks.

Index Terms: Speaker verification, metric learning, residual convolutional neural networks, attentive features pooling.

1. Introduction

Classical approaches for automatic speaker verification (ASV) split the problem into two distinct phases: (i) compute low dimensional speaker representations (i.e., features); and (ii) perform binary classification on top of representations learned from enrollment and test utterances. Recently, given their effectiveness in learning representations from data, neural networks have been used to generate alternative embeddings [1, 2] or to perform the task in an end-to-end fashion [3, 4, 5].

A popular training strategy described in the recent neural network based ASV literature relies on supervised learning where the outputs of some intermediate layers are used as low dimensional embeddings for the task at hand [6, 7, 1, 8, 9, 10]. X-vectors [2], for instance, leverage feed-forward neural networks operating in different time scales to compute low-dimensional embeddings from utterances of varying lengths. The model architecture consists of a frame-level component containing 5 dense layers with context provided by adding neighboring frames as inputs. After that, frame-level model outputs are aggregated in what authors refer to as a statistical pooling layer. Concatenated first- and second-order statistics of frame-level outputs are then used as input for a segment-level model, implemented as a 2 layer feed-forward neural network, finally followed by a softmax output layer. Training is performed for speaker recognition, i.e. the model is used as a classifier aiming to identify the speaker in a given utterance. The softmax layer outputs parametrize a conditional multinoulli distribution over speakers and parameters are learned via maximum likelihood estimation through minimization of the cross-entropy loss using Stochastic Gradient Descent. Outputs of the segment-level model hidden layers are finally used as embeddings, on top of which a classifier of choice can be trained.

Another relevant body of literature on representation learning for ASV is based on metric learning methods [11]. A representation model maps speech features into a low-dimensional space and training is carried out so as to minimize the distance between embeddings from the same speaker while maximizing the distance between those from different speakers. The metric learning approach was explored in large scale for face verification [12], but training difficulties have been commonly reported [13]. As such, stabilizing strategies have been proposed, such as triplets mining, supervised pretraining, and combination of triplets loss with center- [14] or intra-losses [15]. A large-scale speaker verification setting in which triplet loss is employed along with mining strategies can be found in [4].

Even though the metric learning framework seems to be a good fit to learn speaker-dependent low-dimensional representations, the training difficulties observed when training is performed under this setting (i.e., finding informative sets of negative examples at train time) have to be dealt with. In this contribution, we evaluate the combination of two previously described frameworks to overcome this issue, namely: (i)-speaker recognition, and (ii)-metric learning. Our main goal is to combine the advantages offered by each scheme, i.e., the relative easiness of training under the multi-class classification setting along with the discriminability provided by triplet loss minimization. Evaluation of the proposed training strategy is performed using the triplet-network [16] realized with a well-known residual convolutional architecture for automatic speaker verification. A cross-language setting composed of telephone speech is employed for evaluation of the proposed scheme. Additionally, different pooling strategies, used to aggregate sets of local descriptors into a fixed-dimensional representation space, are compared including simple statistics of high-level representations across the time dimension and more complex learning-based attentive schemes.

The remainder of this work is organized as follows: components of our proposed system and training procedure are detailed in Section 2. Experimental setup, results and discussion appear in Section 3 and conclusions in Section 4.

2. System description

We propose speaker representation models obtained through training artificial neural networks under a combination of speaker recognition, i.e. multi-class classification over a closed

http://dx.doi.org/10.21437/Interspeech.2019-2974

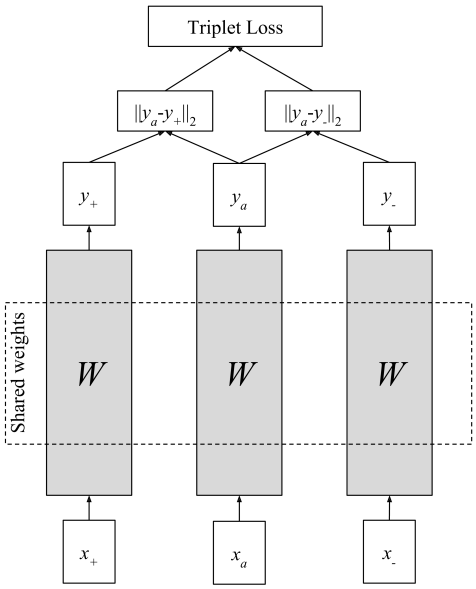

Figure 1: *Illustration of a triplet network setting. Model's weights are shared across the three branches.*

set of speakers, and triplet loss minimization. The latter is performed under the triplet-network setting [16], illustrated in Figure 1 for the case in which a neural network parametrizes a mapping $F: \mathbb{R}^D \rightarrow \mathbb{R}^d$, where D and d correspond to the dimension of data and embeddings, respectively, and $D \gg d$. For a given example $x \in \mathbb{R}^D$, $y = F(x)$ is its counterpart in $\mathbb{R}^d$, and we further assume class labels are assigned to all $x \in \mathbb{R}^D$. Inputs represented by x_a, x_+, and x_- are referred to as anchor, positive, and negative examples, respectively, and compose a triplet such that x_a and x_+ belong to the same class while x_a and x_- are represented by different classes. Parameters W are shared across the three branches, and the triplet loss T [12] can be defined by:

$$T = \max[||y_a - y_+||_2 - ||y_a - y_-||_2 + \alpha, 0], \qquad (1)$$

where the goal is to minimize the euclidean distances between anchor-positive pairs, while maximizing those between anchor-negative pairs. The operator $\max[c, 0]$, $c \in \mathbb{R}$, is used to counter the fact that the $\min[||y_a - y_+||_2]$ is bounded while $\max[||y_a - y_-||_2]$ is not, which could incur in spurious solutions which simply maximize $||y_a - y_-||_2$. α is a hyperparameter commonly referred to as margin.

With the generic triplet-network, we assumed data samples have a fixed dimension D. However, ASV systems often have to deal with recordings of varying lengths. We thus split the mapping from data to embedding into two separate steps: (i) map speech features into a set of vectors representing parts of the input across the time dimension, and (ii) aggregate such local descriptors into a single vector representing the complete input recording. A block diagram describing those two steps are shown in Figure 2. Speech features, namely mel-frequency cepstral coefficients (MFCC), are first fed into an input convolutional layer containing 32 filters of dimension $[C, 3]$, where C is the number of MFCCs in the input, resulting in a 32-channel temporal representation, which then serves as input for the representation model. A standard ResNet-50 [17] is used.

The ResNet's output is a set of N local descriptors $y_i \in \mathbb{R}^K$

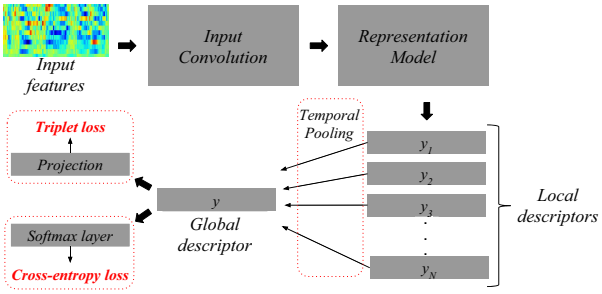

Figure 2: *Block diagram of our proposed system. Features are mapped into local descriptors, aggregated to yield final representations.*

representing parts of the input across time, where N is a function of the input length T and K is the number of filters in the last convolutional layer, set to 512. The following step consists in aggregating the set y_i into what we refer to as a global descriptor, i.e. a vector $y \in \mathbb{R}^K$ representing a speech sample of arbitrary length. We employ three temporal pooling strategies:

Statistics pooling: The global descriptor y will be given by a linear projection of concatenated element-wise estimates of first- and second-order statistics of the set of local descriptors $y_i \in \mathbb{R}^K, i \in \{1, 2, ..., N\}$:

$$y = P \cdot \text{cat}[\mu(y_i), \sigma(y_i)], \qquad (2)$$

where P entries are learned jointly with the representation model parameters. The operator $v = \text{cat}[v_1, v_2]$ concatenates $v_1, v_2 \in \mathbb{R}^d$ such that $v \in \mathbb{R}^{2d}$,, and P is such that the dimension of y is 256.

Attentive pooling: We augment the previous pooling scheme with a weighing method often referred as self-attention. A linear transformation A, whose entries are learned along with the complete model, is first applied to each local descriptor y_i, resulting in the set of scalars $a_{1:N}$:

$$a_i = \tanh(A \cdot y_i). \qquad (3)$$

A set of normalized weights summing up to 1 is then obtained through the softmax operator:

$$w_i = \frac{e^{a_i}}{\sum_{i=1}^{N} e^{a_i}}, \qquad (4)$$

and the global descriptor y is finally given by the projection of concatenated statistics of weighted local descriptors, i.e.,:

$$y = P \cdot \text{cat}[\mu(w_i \cdot y_i), \sigma(w_i \cdot y_i)]. \qquad (5)$$

Recurrent attentive pooling: Additionally, recurrent layers are applied along with self-attention scheme described above so that its hidden layer can be further used as a summarization of the set of local descriptors. The recurrent model is implemented as a two-layered bi-directional LSTM [18] with hidden layer set to a size of 256. The LSTM will first map the set y_i into a new sequence y'_i and a hidden state h, hence y becomes:

$$y = P \cdot \text{cat}[\mu(w_i \cdot y'_i), \sigma(w_i \cdot y'_i), h], \qquad (6)$$

where weights w_i are obtained with self-attention on y'_i.

Once the embedding y is obtained through one of the pooling approaches described, our two loss components can be computed. Triplet loss will be computed on top of embedings projected on the unit sphere: $\frac{y}{||y||}$. For the speaker recognition

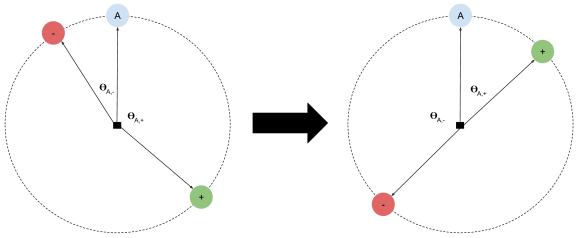

Figure 3: *Hard (left) and easy (right) triplets. Triplet loss minimizes the distance between anchor and positive examples while maximizing that between anchor and negative cases.*

term, a softmax output layer is employed so that the multi-class cross entropy can be computed using speaker IDs as class labels. Training is performed so as to minimize the sum of losses.

2.1. Minibatch construction and triplets selection

Minibatches are constructed through sequentially sampling examples from each speaker. More specifically, 5 recordings are sampled with repetition and further processed such that a randomly positioned window of 10 seconds will be selected for those longer than 10 seconds, and shorter-duration training examples are elongated by repeating initial frames so as to make them reach a minimum of 10 seconds duration. To further increase the diversity on training samples, each minibatch has its length randomly selected to lie between 3-10 seconds before being fed into the model. Minibatches of size $24 \cdot 5$ are used throughout training. At test time, however, recordings are fed with their original length. One training epoch is considered finished when sets of 5 recordings are sampled from each speaker 3 times. A budget of 500 epochs is used for each training run.

Previous literature has discussed in depth the need of finding sets of triplets yielding a high triplet loss, i.e. *hard* triplets in the sense that $||y_a - y_-||_2 < ||y_a - y_+||_2$ resulting in an informative T. In [13], authors argue representation models are able to quickly learn to map trivial triplets correctly, which renders a large fraction of possible triplets uninformative. We illustrate *hard* and *easy* examples in Figure 3. We thus employ an online triplet selection scheme similar to that introduced in [12] and select hard positive/negative examples from within a given minibatch embedded by the most current version of the model. We first create all possible positive pairs. The hardest negative pairs are then selected to match the number of positive pairs.

2.2. Maximum Entropy Regularization

Given that our embeddings or global descriptors y lie in the unit sphere in $\mathbb{R}^K$, we employ an entropy regularizer so as to enforce speaker representations to spread across the sphere, which favors discriminability. A regularization penalty was introduced in [19] specifically to this end based on the Kozachenko-Leononenko estimator of the differential entropy [20]. For a finite sample of size n, we use a slightly modified maximum entropy regularizer [19] for embeddings $y_j \in \mathbb{R}^K$:

$$H_n = -\sum_{j=1}^{n} \rho_{n,j}, \tag{7}$$

where $\rho_{n,j} = \min_{j \neq k} ||y_j - y_k||_2$. A coefficient λ is further added in order to control the influence of the regularizer. We set λ to $1\mathrm{e}{-2}$ in all our experiments.

As a practical remark, we found such regularization penalty to alleviate numerical instabilities observed when training probabilistic linear discriminant analysis (PLDA) [21] on top of embeddings obtained using our systems. Global descriptors extracted with earlier versions of the setting described here would very often yield design matrices of incomplete rank, which is not the case when representation models are encouraged to spread embeddings over the sphere.

2.3. Training details

Training is performed with Stochastic Gradient Descent and weight decay is further employed as a regularization penalty. For the initial learning rate and weight decay coefficient, we found the values of $1\mathrm{e}{-2}$ and $5\mathrm{e}{-5}$ to yield the best results across considered settings. Polyak's acceleration is also employed with its coefficient set at the default value of 0.9. A schedule is defined such that the learning rate is reduced by a factor of 0.5 once a validation loss does not improve for at least 15 epochs. A validation set is built by selecting all the recordings of a group of 50 randomly selected speakers, taken out of training data. The validation loss is measured as equal error rate (EER) using cosine similarity to score a set of trials created at train time out of recordings from the validation set. Training takes approximately 8 days to complete in a single NVIDIA Titan X GPU[1], and results are reported for the model that achieved the lowest validation EER during training.

3. Experiments

Experiments are carried out using the data introduced for the the cross-language *NIST SRE* 2016 evaluation corresponding to test recordings in Tagalog and Cantonese. Train data is composed of *NIST SRE*s from 2004 to 2010 combined with *Mixer* 6 as well as *Switchboard*-2, phases 1, 2, and 3, summing up to approximately 7000 speakers, out of which we remove all the recordings from 50 speakers to be used as validation set. Training is performed on top of 23 MFCCs. We further introduce multi-condition training data by augmenting the original train partition with supplementary noisy speech, created by corrupting original samples adding reverberation (reverberation time varies from 0.25s - 0.75s), as well as by adding background noise such as music (signal-to-noise ratio, SNR, within 5-15dB), and babble (SNR varies from 10 to 20dB). Noise signals were selected from the MUSAN corpus [22] and the room impulse responses to simulate reverberation from [23].

PLDA was employed for scoring trials after dimensionality reduction of embeddings from 256 to 80 using linear discriminant analysis (LDA). PLDA is trained on embeddings from the *SRE* partition of the training data, which are computed following the same approach as described for test data. In order to overcome the domain shift between train and test data due to different spoken languages, the model adaptation scheme introduced in [24] is utilized for PLDA. To do so, embeddings of unlabelled data in Cantonese and Tagalog are clustered, and clusters are used as speaker identities, which are then employed for training a second PLDA model. The final model is obtained by simply averaging the second order statistics of the two trained models. All reported results are obtained from the models that achieved the minimal validation loss throughout training, and the evaluation data is only made available to the models to generate the reported metrics, not being used at development phase.

[1]Code is available at: `https://github.com/joaomonteirof/multitask_asv`

Table 1: *EER obtained for the same system trained with different losses. The combination of triplet loss and cross-entropy yield speaker-dependent representations.*

Backend	*Train loss*	*All*	*Cantonese*	*Tagalog*
PLDA	Triplet loss	29.51%	27.25%	31.94%
	Cross-entropy	20.81%	17.01%	24.72%
	Combined	14.10%	9.23%	19.09%
Adapted PLDA	Triplet loss	26.97%	24.47%	29.48%
	Cross-entropy	18.81%	14.52%	22.57%
	Combined	**10.51%**	**6.44%**	**14.67%**

Table 2: *EER obtained using different pooling strategies to aggregate local descriptors into embeddings.*

Backend	*Pooling*	*All*	*Cantonese*	*Tagalog*
PLDA	Statistics	14.07%	8.95%	19.14%
	Attention	14.10%	9.23%	19.09%
	LSTM	15.99%	11.93%	19.56%
	Pretrain+LSTM	**13.29%**	**8.42%**	**18.22%**
Adapted PLDA	Statistics	10.87%	6.79%	15.00%
	Attention	10.51%	6.44%	14.67%
	LSTM	10.57%	6.57%	14.65%
	Pretrain+LSTM	**9.93%**	**6.07%**	**13.86%**

We start our evaluation with an ablation study by comparing one of our proposed models trained with the combined loss with similar systems trained with cross-entropy or triplet losses only. Self-attention was the pooling strategy used for the three models. Results in terms of equal error rate (EER) are shown in Table 1. EER consists of the value of the false acceptance rate at the score threshold in which it matches the false rejection rate. Verification performance is significantly improved with respect to systems trained with only one of the losses for both languages and backends considered, which supports the claim that both strategies are complimentary and should be used jointly.

In Table 2 we compare the verification performance of the three pooling strategies considered in this work in order to map the set of local descriptors into an embedding. We further evaluate a fourth system obtained by first training with self-attention and then fine-tuning the model after including the LSTM block in the pooling. Interestingly, the different pooling methods benefit differently from PLDA adaptation and the rank of best performers, not considering the pretrained model, changes considerably once adaptation is used. This finding indicates the extra capacity in terms of number of parameters at the pooling level is used to learn domain-dependent patterns, and thus the performance boost achieved with adaptation is higher in such cases. Regarding pretraining, its benefit is observed both with and without adaptation, yielding the lowest ERR obtained from single systems within our evaluation.

We further perform a comparison among our proposed method and well-known systems. We follow the setting in [25] and i-vectors [26] are obtained by computing a 2048-Gaussians full covariance universal background model (UBM) using the unlabelled partition of *NIST SRE* 2016. Total variability analysis is performed on top of UBM's Baum-Welch statistics obtained for recordings from the *SRE* partition of training data, resulting in a 600-dimensional extractor; i-vectors are finally reduced to a dimension of 200 with LDA. An x-vector system is also obtained using its Kaldi recipe [2] thus yielding embeddings of dimension 512 later reduced to 150 with LDA. The same train data as our systems was used fo training x-vectors. Scoring of both i- and x-vectors is performed with PLDA and adapted PLDA. We only report EERs obtained with adapted PLDA for those cases given the lower EERs achieved.

Table 3: *Comparison of proposed systems with well-known baseline methods. Results correspond to verification EER.*

	System	*All*	*Cantonese*	*Tagalog*
Baseline	x-vector [8]	11.90%	6.50%	16.30%
	x-vector+Att. [28]	10.21%	4.61%	14.15%
	x-vector	10.02%	5.82%	14.31%
	i-vector	12.68%	8.17%	17.25%
Proposed	Statistics	10.87%	6.79%	15.00%
	Attention	10.51%	6.44%	14.67%
	LSTM	10.57%	6.57%	14.65%
	Pretrain+LSTM	**9.93%**	**6.07%**	**13.86%**
Fusion	Proposed	8.50%	5.00%	12.10%
	Proposed + x-vectors	7.92%	4.48%	11.48%
	Proposed + x-vec + i-vec	**7.73%**	**4.33%**	**11.19%**

Results in Table 3 correspond to: (i) a set of baseline systems on the first four rows given by EERs obtained by above described i- and x-vectors and further results reported in the x-vector original paper [8] as well as in a more recent work in which x-vectors were augmented with structured self-attention [27]. (ii) our proposed systems in the mid rows corresponding to varying time pooling strategies. EERs are reported only for adapted PLDA in this case, and (iii) sum fusion performed at the score level. We report results obtained by fusing scores of proposed systems only, as well as proposed combined with x-vectors, and also i-vectors. As can be seen, pretrained system attains the lowest EER among single systems for the case of the pooled set of trials, i.e. complete set of trials containing both Cantonese and Tagalog recordings. This is due to the improvement observed in this system for the Tagalog case, since some of the considered baselines are able to achieve a lower EER in the particular case of evaluation on Cantonese trials only. By fusing the considered systems we observe further improvement, thus reaching the lowest EER across all evaluation conditions.

4. Conclusion

We introduced a training scheme with the goal of obtaining speaker dependent representations from speech of unconstrained phonetic content. A combination of triplet loss and multi-class cross-entropy is used as a training signal for a neural network consisting of a ResNet-based representation model. The model maps a recording of arbitrary length into a set of local descriptors, and a pooling block is responsible for aggregating such descriptors into a single embedding. Evaluation of the proposed method is performed on the cross-language *NIST SRE* 2016 in which we show that: (i) models trained with the combined loss outperform those trained with single losses, and (ii) our best system yields relevant improvements in verification performance relative to well-known benchmark methods. Score-level sum fusion among our systems, as well as our systems and baselines, further boosted performance.

5. Acknowledgements

The authors wish to acknowledge funding from the Natural Sciences and Engineering Research Council of Canada (NSERC) through grant RGPIN-2019-05381. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the NSERC.

6. References

[1] G. Bhattacharya, J. Alam, P. Kenn, and V. Gupta, "Modelling speaker and channel variability using deep neural networks for robust speaker verification," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 192–198.

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

[3] J. Rohdin, A. Silnova, M. Diez, O. Plchot, P. Matejka, and L. Burget, "End-to-end DNN Based Speaker Recognition Inspired by i-vector and PLDA," *ArXiv e-prints*, Oct. 2017.

[4] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep Speaker: an End-to-End Neural Speaker Embedding System," *ArXiv e-prints*, May 2017.

[5] 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.

[6] G. Bhattacharya, J. Alam, and P. Kenny, "Deep speaker embeddings for short-duration speaker verification," in *Proc. Interspeech 2017*, 2017, pp. 1517–1521.

[7] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, "Deep Speaker Feature Learning for Text-independent Speaker Verification," *ArXiv e-prints*, May 2017.

[8] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," *Proc. Interspeech 2017*, pp. 999–1003, 2017.

[9] A. Torfi, J. Dawson, and N. M. Nasrabadi, "Text-Independent Speaker Verification Using 3D Convolutional Neural Networks," *ArXiv e-prints*, May 2017.

[10] S. Novoselov, A. Shulipa, I. Kremnev, A. Kozlov, and V. Shchemelinin, "On deep speaker embeddings for text-independent speaker recognition," *ArXiv e-prints*, Apr. 2018.

[11] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proceeedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2006, pp. 1735–1742.

[12] 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.

[13] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017.

[14] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European Conference on Computer Vision*. Springer, 2016, pp. 499–515.

[15] N. Le and J.-M. Odobez, "Robust and discriminative speaker embedding via intra-class distance variance regularization," *Proc. Interspeech 2018*, pp. 2257–2261, 2018.

[16] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*. Springer, 2015, pp. 84–92.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] A. Sablayrolles, M. Douze, C. Schmid, and H. Jégou, "Spreading vectors for similarity search," 2018.

[20] J. Beirlant, E. J. Dudewicz, L. Györfi, and E. C. Van der Meulen, "Nonparametric entropy estimation: An overview," *International Journal of Mathematical and Statistical Sciences*, vol. 6, no. 1, pp. 17–39, 1997.

[21] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[22] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015, arXiv:1510.08484v1.

[23] "Open Speech and Language Resources," 2017, http://www.openslr.org/28/.

[24] D. Garcia-Romero, A. McCree, S. Shum, N. Brummer, and C. Vaquero, "Unsupervised domain adaptation for i-vector speaker recognition," in *Proceedings of Odyssey: The Speaker and Language Recognition Workshop*, 2014.

[25] J. Alam, G. Bhattacharya, and P. Kenny, "Speaker verification in mismatched conditions with frustratingly easy domain adaptation," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 176–180.

[26] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[27] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," *arXiv preprint arXiv:1703.03130*, 2017.

[28] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey, "Self-attentive speaker embeddings for text-independent speaker verification," in *Proc. Interspeech*, 2018, pp. 3573–3577.