IberSPEECH 2021
24-25 March 2021, Valladolid, Spain

Self-supervised Deep Learning Approaches to Speaker Recognition: A Ph.D. Thesis Overview

Umair Khan and Javier Hernando

TALP Research Center, Department of Signal Theory and Communications,
Universitat Politecnica de Catalunya Barcelona, Spain
{umair.khan, javier.hernando}@upc.edu

Abstract

Recent advances in Deep Learning (DL) for speaker recognition have improved the performance but are constrained to the need of labels for the background data, which is difficult in practice. In i-vector based speaker recognition, cosine (unsupervised) and PLDA (supervised) are the basic scoring techniques, with a big performance gap between the two. In this thesis we tried to fill this gap without using speaker labels in several ways. We applied Restricted Boltzmann Machine (RBM) vectors for the tasks of speaker clustering and tracking in TV broadcast shows. The experiments on AGORA database show that using this approach we gain a relative improvement of 12% and 11% for speaker clustering and tracking tasks, respectively. We also applied DL techniques in order to increase the discriminative power of i-vectors in speaker verification task, for which we have proposed the use of autoencoder in several ways, i.e., (1) as a pre-training for a Deep Neural Network (DNN), (2) as a nearest neighbor autoencoder for i-vectors, (3) as an average pooled nearest neighbor autoencoder. The experiments on VoxCeleb database show that we gain a relative improvement of 21%, 42% and 53%, using the three systems respectively. Finally we also proposed a self-supervised end-to-end speaker verification system. The architecture is based on a Convolutional Neural Network (CNN), trained as a siamese network with multiple branches. From the results we can see that our system shows comparable performance to a supervised baseline.

Index Terms: deep learning, speaker verification, i-vector, autoencoder, CNN, speaker embeddings

1. Introduction

Deep Learning (DL) approaches have shown their success in image and speech technologies which has inspired the community to apply these approaches in speaker recognition as well [1, 2, 3]. The current DL application in speaker recognition can be categorized as: at the frontend, like [4, 5, 6, 7, 8, 9], at the backend, such as in [10, 11, 12], and as an end-to-end system such as in [13, 14, 15]. The most common and the so called speaker embeddings are typically extracted from an intermediate layer of a Deep Neural Network (DNN). The inputs to the network are feature vectors, like the Mel-Frequency Cepstral Coefficients (MFCC) or in some cases spectrograms. Whereas the output of the network is fed with the class (speaker) labels for the background data. Therefore, these DL approaches are typically constrained to labeled background data.

The i-vector representation of speech [16], with cosine scoring, is an unsupervised process. However, Probabilistic Linear Discriminant Analysis (PLDA) [17] is the most efficient backend for i-vectors which leads to a superior performance as compared to cosine scoring but at the cost of labeled background data. However, in practice, it is difficult to access large amount of labeled data. In this thesis [18], we applied self-supervised DL approaches to improve the performance without using speaker labels. We addressed this problem in three different ways.

As a first objective of this thesis, we applied Restricted Boltzmann Machine (RBM) vector representation of speech for the tasks of speaker clustering and tracking in TV broadcast shows. Such a representation is referred to as RBM vector which has shown success in speaker verification task in [19]. In the second objective, we applied self-supervised DL approaches in order to increase the discriminative power of i-vectors for speaker verification. For this purpose we used autoencoder in three different ways, i.e., (1) as a pre-training for a DNN, (2) as a nearest neighbor autoencoder for i-vectors, (3) as an average pooled nearest neighbor autoencoder. In the last main objective of this thesis, we proposed a self-supervised end-to-end speaker verification system. The network architecture is based on a Convolutional Neural Network (CNN) which is trained as a siamese network with multiple branches.

The rest of the paper is organized as follows. Sections 2, 3, and 4 explain the three main objectives of the thesis, respectively. Section 5 describes the experimental setup and results. Section 6 lists the publication resulted from the Ph.D. thesis and section 7 concludes the paper.

2. RBM vectors for speaker clustering and tracking

We have proposed RBMs at the front-end for the tasks of speaker clustering and speaker tracking in TV broadcast shows. RBMs are trained to transform utterances into a vector based representation. Because of the lack of data for a test speaker, we propose RBM adaptation to a global model. First, the speaker independent global model, which is referred to as universal RBM (URBM), is trained with all the available background data. Then a speaker dependent adapted RBM model is trained with the data of each test speaker. The visible to hidden weight matrices of the adapted models are concatenated along with the bias vectors and are whitened using Principal Component Analysis (PCA) to generate the vector representation of speakers. These vectors, referred to as RBM vectors, were shown to preserve speaker-specific information and are used in the tasks of speaker clustering and speaker tracking. Figure 1 shows a visualization of the connection weights of the URBM (top) and of two randomly selected speakers (bottom). From the figure, it is clear that the speaker dependent adapted RBM weights are driven in speaker-specific direction which are discriminative.

For the speaker clustering task, we extract RBM vectors using the method described above for the test speakers. All the speaker segments that are to be clustered are represented by

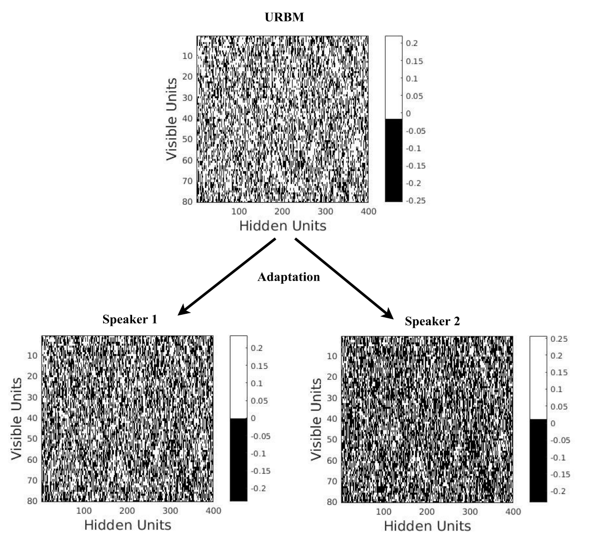

Figure 1: *Comparison of URBM and adapted RBMs weights.*

RBM vectors. Then we cluster these RBM vectors by applying a bottom-up Agglomerative Hierarchical Clustering (AHC) approach using cosine and PLDA scores [20]. For the speaker tracking task, we implement a two stage strategy. The first stage is speaker segmentation in which the audio is segmented according to the speaker change points [21]. In the second stage, the segments generated are identified against all the target speakers, in order to specify which segment belongs to which target speaker [21, 22]. We represent all the segments and target speakers by RBM vectors. Then, the RBM vectors of all the segments are scored against the RBM vectors of all the target speakers using cosine and PLDA scoring.

3. Autoencoder based approaches for speaker verification

3.1. Autoencoder as a pre-training for DNN

In this work we have proposed the use of autoencoder pre-training for post-processing of i-vectors in speaker verification task. The conventional architecture of an autoencoder consists of an $encoder$ and a $decoder$ as shown in Figure 2 (left). The $encoder$ is a function that encodes the input i-vector w into a lower dimensional space, and the $decoder$ is a function that decodes it back in order to reconstruct w. In order to avoid the need of large amount of labeled background data, we train the autoencoder using a large amount of unlabeled data. The training is carried out by minimizing the Mean Square Error (MSE) between the input w and the reconstructed $w\hat{}$. Then, we train a DNN classifier using a relatively small labeled data. Therefore, this is a semi-supervised DL approach. We initialize the parameters of the DNN training with the weight matrices and bias vectors of the pre-trained autoencoder. In this way, we train a hybrid autoencoder-DNN classifier. After the training, we transform i-vectors into new representation as the output from the second last layer of the network as shown in Figure 2. The goal is to improve the performance using fewer background speaker labels. The experimental results have shown that the proposed approach has improved the baseline system in two aspects. Firstly, the proposed system outperforms the baseline system in terms of EER. Secondly, we have observed

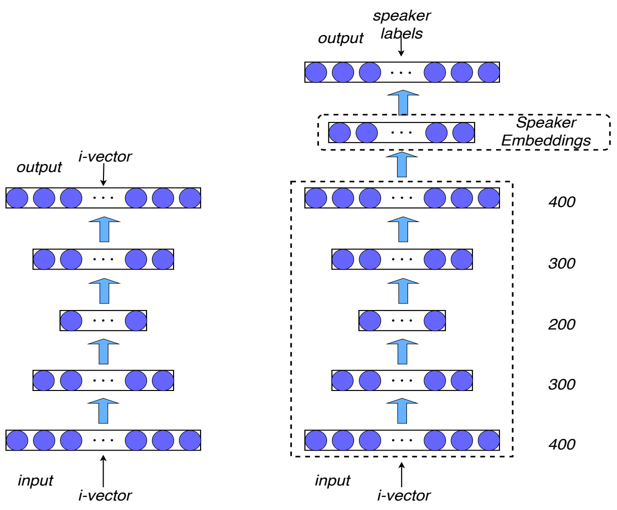

Figure 2: *(left) Autoencoder pre-training (right) DNN training.*

that the hybrid autoencoder-DNN training converges faster as compared to the one without autoencoder pre-training [23].

3.2. Nearest neighbor approach

In this work we have proposed to train an autoencoder to reconstruct neighbor i-vectors instead of the same training i-vectors, as usual. These neighbor i-vectors are selected in an unsupervised manner according to the highest cosine scores to the training i-vectors. In this way the autoencoder learns speaker variability when no labeled background data is available. The conventional training is carried out by minimizing the loss function : $MSE(w\hat{}, w)$. We propose to train the autoencoder by minimizing the loss function : $MSE(w\hat{}, v)$, as shown in Figure 3, where v is a neighbor i-vector of w and $w\hat{} = decoder(encoder(w))$. We propose an automatic selection of the neighbor i-vectors according to the Algorithm 1, as explained in [24].

Once the autoencoder is trained with the selected neighbor i-vectors, we transform the testing i-vectors into a new speaker vector representation. We extract the desired speaker vectors at the output of the autoencoder. These are referred to as autoencoder vectors or shortly ae-vectors. In the experiments, ae-vectors have shown to increase the discriminative quality of i-vectors without using speaker labels.

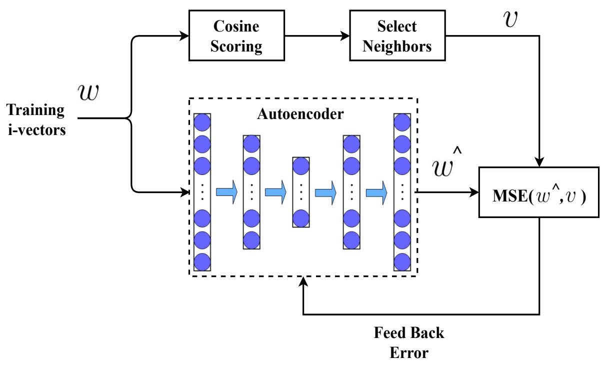

Figure 3: *Proposed training of the autoencoder.*

Algorithm 1: Proposed neighbor i-vectors selection algorithm for a constant k

```
Input  : Training i-vectors $w_i$, $1 < i < n$
Output: Neighbor i-vectors $v_{ij}$, $1 < i < n$ and
        $1 < j < k$
1 for each training i-vector $w_i$ do
2     for each training i-vector $w_t$, $1 < t < n$ do
3         if $i \neq t$ then
4             Compute $score_{i,t} = cosine(w_i, w_t)$
5         end
6     end
7     Select the corresponding $k$ i-vectors with the
        highest scores as $v_{i,j}$
8 end
```

3.3. Average pooled nearest neighbor approach

In this work we train the network using a large set of nearest neighbor i-vectors. For every input i-vector w a set of neighbor i-vectors v_k is input to the network. During the training we minimize the loss between reconstructed $\hat{v}$ and actual training i-vector w. The neighbor i-vectors are selected using the same algorithm but setting a $threshold$ after selecting the constant k number of neighbors. The network architecture is composed of an average pooling layer followed by four fully connected (FC) layers. The input layer is fed by the set of neighbor i-vectors v_k. We train the network by minimizing the loss function $L(\hat{v}, w)$, where $L(\cdot)$ can be Cosine Distance (CD) or MSE, w is the training i-vector and $\hat{v} = f(v_k)$, where $f(\cdot)$ is the non-linearity deployed by the network. During the training, the loss $L(\cdot)$ is back-propagated to the network in every iteration. In this way, the DNN is able to learn from the nearest neighbor i-vectors and avoids using actual speaker labels. After training, we extract speaker vectors for the testing i-vectors, which are used in the experiments [25].

4. End-to-end system

In this work we propose self-supervised siamese networks trained using pairwise training samples, i.e., anchor, client and impostor. Since we do not use speaker labels, we propose to generate the training pairs in an unsupervised manner. The client and impostor selection is carried out in the i-vector space using two databases, i.e., A and B. Suppose Spk_A and Spk_B denote the speakers appearing in database A and B, respectively. We assume that the speakers in database A do not appear in database B, i.e., $Spk_A \cap Spk_B = \phi$.

First, all the i-vectors in A are scored among each other using cosine scoring. For every i-vector in A we select a fix k number of neighbor i-vectors using Algorithm 1 as client i-vectors. After this we apply a $threshold$ to the cosine scores. Then we score all the i-vectors in A with those in B. For every i-vector in A, we select k number of i-vectors from B that are closest according to scores. As the speakers in A do not appear in B, these k selected i-vectors, subjected to a $threshold$, are the impostors i-vectors. In this way, every i-vector in A has been assigned k client and k impostor i-vectors. The network has two identical branches and is trained by minimizing binary cross-entropy loss as shown in Figure 4. The training pairs are in the form [anchor, client] and [anchor, impostor]. After training, we obtain decision scores for the experimental trials at the output of the network. We also trained a triple-branch siamese for which

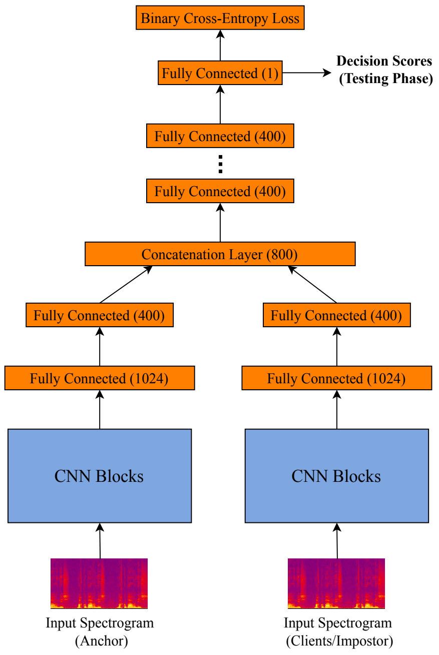

Figure 4: *Block diagram of our end-to-end siamese network.*

we make pairs of three samples, i.e., [anchor, client, impostor], and is trained by minimizing triplet loss. Unlike the double-branch siamese, we extract speaker embeddings from the triple-branch siamese [26].

5. Experimental results

For speaker clustering and tracking experiments we used AGORA Catalan broadcast TV3 dataset [27]. We extracted 2631 speaker segments from the test partition, according to the transcription, for speaker clustering. 414 target speakers were tracked during the tracking experiments. Table 1 shows the results obtained using RBM vectors compared with i-vectors. For speaker clustering an Equal Impurity (EI) of 37.14% is obtained with 2000 dimensional RBM vectors using cosine scoring, gaining a relative improvement of 12% over i-vectors. Similarly an EI of 31.68% is obtained with PLDA scoring which results in a relative improvement of 11% over i-vectors. For speaker tracking Equal Error Rate (EER) of 3.30% and 2.74% were obtained using 2000 dimensional RBM vector using cosine and PLDA scoring respectively, with relative improvements of 11.76% and 7.74% over baseline.

Table 1: *Speaker clustering (EI) and tracking (EER) results.*

Clustering	**EI (Cosine)**	**EI (PLDA)**
[1] i-vector (400)	46.26	36.16
[2] i-vector (800)	42.19	35.91
[3] i-vector (2000)	42.83	35.89
[4] RBM vector (400)	39.66	37.36
[5] RBM vector (800)	40.02	32.36
[6] RBM vector (2000)	37.14	31.68
Tracking	**EER (Cosine)**	**EER (PLDA)**
[7] i-vector (800)	3.74	2.97
[8] RBM vector (2000)	3.30	2.74

Table 2: *Speaker verification results using autoencoder pre-training for DNN, in terms of EER.*

Approach	Scoring	EER(%)
[1] i-vector	Cosine	17.61
[2] i-vector	PLDA	9.54
[3] only-encoder-dnn	Cosine	12.73
[4] conventional-dnn	Cosine	8.58
[5] full-autoencoder-dnn	Cosine	7.51

Table 3: *Speaker verification results using nearest neighbor approach, for different values of k, using cosine scoring.*

Approach	k	EER(%)
[1] i-vector	-	17.61
[2] ae-vector	1	15.32
[3] ae-vector	2	12.36
[4] ae-vector	5	10.62
[5] ae-vector	15	10.20
Fusion of [1] & [5]	-	9.82

For speaker verification we used VoxCeleb-1 [28] dataset for the nearest neighbor and average pooled nearest neighbor approaches, and VoxCeleb-2 [29] dataset for the autoencoder pre-training and end-to-end systems. From the test partition of VoxCeleb-1, 37,720 speaker verification trials were scored for evaluation. Table 2 compares the performance of the autoencoder pre-training with i-vectors. From the table it is clear that our proposed speaker vectors has outperformed the i-vector system by a relative improvement of 21%, in terms of EER. Also, it is shown in [23] that using the pre-training strategy the DNN training convergence was faster than the conventional training. Table 3 shows the results obtained using the nearest neighbor approach for different values of k. From the table we observe that the ae-vectors has gained a relative improvement of 42% over i-vector/cosine. Moreover the EER of 10% is very close to that of i-vector/PLDA (9.54%). Table 4 shows the results obtained using the average pooled nearest neighbor approach for different values of k using CD and MSE losses. We observed that MSE loss is the best choice. Moreover our approach has gained a relative improvement of 53% over i-vector/PLDA at the cost of using background data in testing [25].

Table 5 shows the results obtained using the end-to-end and triple-branch networks. From the Table we can see that as we increase the value of k, the performance improves. The best EER of 6.90% was achieved using k equal to 10. Setting the value of k equal to 10, we have trained our triple-branch siamese network using triplet loss and we extracted speaker embeddings. The triple-branch siamese network has achieved an EER of 6.95%. A score level fusion gives an EER of 6.07% which is very close to the supervised AMSoftmax baseline [26, 30].

6. Publications

[1] Umair Khan, Pooyan Safari, and Javier Hernando. Restricted Boltzmann Machine Vectors for Speaker Clustering. In *Proc. IberSPEECH*, pages 10–14, 2018, **(Awarded the ISCA travel grant).**

[2] Umair Khan, Pooyan Safari, and Javier Hernando. Restricted boltzmann machine vectors for speaker clustering and tracking tasks in tv broadcast shows. *Applied Sciences*, 9(13):2761, 2019.

Table 4: *Speaker verification results using average pooled nearest neighbor approach with CD and MSE, using cosine scoring.*

k	CD Loss	MSE Loss
[1] 10	8.81	8.70
[2] 20	6.60	6.56
[3] 30	5.68	5.64
[4] 50	4.97	4.98
[5] 100	4.84	4.45
[6] 150	6.53	4.48

Table 5: *Speaker verification results using end-to-end and triple-branch siamese, in comparison to supervised baselines.*

Approach	k	EER(%)
[1] i-vector/PLDA	-	9.54
[2] Baseline (AMSoftmax)	-	5.71
[3] End-to-end	2	7.81
[4] End-to-end	5	7.73
[5] End-to-end	10	6.90
[6] Triple-branch	10	6.95
Fusion of [5] & [6]	10	6.07

[3] Umair Khan and Javier Hernando. Dnn speaker embeddings using autoencoder pre-training. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE, 2019.

[4] Umair Khan, Miquel India, and Javier Hernando. Auto-encoding nearest neighbor i-vectors for speaker verification. *Proc. Interspeech 2019*, pages 4060–4064, 2019.

[5] Umair Khan, Miquel India, and Javier Hernando. i-vector transformation using k-nearest neighbors for speaker verification. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 7574–7578. IEEE, 2020.

[6] Umair Khan and Javier Hernando. Unsupervised training of siamese networks for speaker verification. *Proc. Interspeech 2020*, pages 3002–3006, 2020.

[7] Umair Khan and Javier Hernando. The upc speaker verification system submitted to voxceleb speaker recognition challenge 2020 (voxsrc-20). *arXiv preprint arXiv:2010.10937*, 2020, **(3rd prize winner of self-supervised track).**

7. Conclusions

The contributions of this thesis are presented in three main objectives. Firstly, the use of RBM vectors for speaker clustering and tracking, which resulted in a relative improvement (RI) of 12% and 11%, respectively. Secondly, we applied DL techniques to improve i-vectors for speaker verification, in several ways. The experimental results show that we gain a RI of 21%, 42% and 53%. Finally we trained an end-to-end speaker verification system, which showed a comparable performance to supervised baseline. From the thesis we conclude that using DL approaches, despite of being unsupervised, we could do better in scenarios where labels are not available for the training data.

8. Acknowledgements

This work was supported by the project PID2019-107579RB-I00 / AEI / 10.13039/501100011033

9. References

[1] K. Chen and A. Salman, "Learning speaker-specific characteristics with a deep neural architecture," *IEEE Transactions on Neural Networks*, vol. 22, no. 11, pp. 1744–1756, 11 2011.

[2] F. Richardson, D. Reynolds, and N. Dehak, "Deep neural network approaches to speaker and language recognition," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671–1675, 10 2015.

[3] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, "Deep neural networks for extracting baum-welch statistics for speaker recognition," in *Proc. Odyssey*, 2014, pp. 293–298.

[4] Y. Liu, Y. Qian, N. Chen, T. Fu, Y. Zhang, and K. Yu, "Deep feature for text-dependent speaker verification," *Speech Communication*, vol. 73, pp. 1–13, 2015.

[5] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4052–4056.

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

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

[8] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," *Proc. Interspeech 2018*, pp. 2252–2256, 2018.

[9] S. Novoselov, A. Shulipa, I. Kremnev, A. Kozlov, and V. Shchemelinin, "On deep speaker embeddings for text-independent speaker recognition," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 378–385.

[10] M. Senoussaoui, N. Dehak, P. Kenny, R. Dehak, and P. Dumouchel, "First attempt of boltzmann machines for speaker verification," in *Odyssey 2012-The Speaker and Language Recognition Workshop*, 2012.

[11] T. Stafylakis, P. Kenny, M. Senoussaoui, and P. Dumouchel, "Plda using gaussian restricted boltzmann machines with application to speaker verification," in *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[12] O. Ghahabi and J. Hernando, "Deep learning backend for single and multisession i-vector speaker recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 4, pp. 807–817, 4 2017.

[13] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[14] C. Zhang and K. Koishida, "End-to-end text-independent speaker verification with triplet loss on short utterances." in *Interspeech*, 2017, pp. 1487–1491.

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

[16] 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.

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

[18] U. Khan, "Self-supervised deep learning approaches to speaker recognition," Ph.D. dissertation, Universitat Politècnica de Catalunya, 2021.

[19] P. Safari, O. Ghahabi, and J. Hernando, "From features to speaker vectors by means of restricted boltzmann machine adaptation," in *ODYSSEY 2016-The Speaker and Language Recognition Workshop*, 2016, pp. 366–371.

[20] U. Khan, P. Safari, and J. Hernando, "Restricted Boltzmann Machine Vectors for Speaker Clustering," in *Proc. IberSPEECH*, 2018, pp. 10–14.

[21] U. Khan, "Speaker tracking system using speaker boundary detection," Master's thesis, Universitat Politècnica de Catalunya, 2016.

[22] U. Khan, P. Safari, and J. Hernando, "Restricted boltzmann machine vectors for speaker clustering and tracking tasks in tv broadcast shows," *Applied Sciences*, vol. 9, no. 13, p. 2761, 2019.

[23] U. Khan and J. Hernando, "Dnn speaker embeddings using autoencoder pre-training," in *2019 27th European Signal Processing Conference (EUSIPCO)*. IEEE, 2019, pp. 1–5.

[24] U. Khan, M. India, and J. Hernando, "Auto-encoding nearest neighbor i-vectors for speaker verification," *Proc. Interspeech 2019*, pp. 4060–4064, 2019.

[25] U. Khan, M. India, and J. Hernando, "I-vector transformation using k-nearest neighbors for speaker verification," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7574–7578.

[26] U. Khan and J. Hernando, "Unsupervised training of siamese networks for speaker verification," *Proc. Interspeech 2020*, pp. 3002–3006, 2020.

[27] H. Schulz and J. A. R. Fonollosa, "A catalan broadcast conversational speech database," in *Joint SIG-IL/Microsoft Workshop on Speech and Language Technologies for Iberian Languages*, 2009, pp. 27–30.

[28] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Interspeech*, 2017.

[29] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Interspeech*, 2018.

[30] U. Khan and J. Hernando, "The upc speaker verification system submitted to voxceleb speaker recognition challenge 2020 (voxsrc-20)," *arXiv preprint arXiv:2010.10937*, 2020.