Deep Self-Supervised Hierarchical Clustering for Speaker Diarization

Prachi Singh, Sriram Ganapathy

Learning and Extraction of Acoustic Patterns (LEAP) Lab, Indian Institute of Science.

`prachisingh@iisc.ac.in, sriramg@iisc.ac.in`

Abstract

The state-of-the-art speaker diarization systems use agglomerative hierarchical clustering (AHC) which performs the clustering of previously learned neural embeddings. While the clustering approach attempts to identify speaker clusters, the AHC algorithm does not involve any further learning. In this paper, we propose a novel algorithm for hierarchical clustering which combines the speaker clustering along with a representation learning framework. The proposed approach is based on principles of self-supervised learning where the self-supervision is derived from the clustering algorithm. The representation learning network is trained with a regularized triplet loss using the clustering solution at the current step while the clustering algorithm uses the deep embeddings from the representation learning step. By combining the self-supervision based representation learning along with the clustering algorithm, we show that the proposed algorithm improves significantly (29% relative improvement) over the AHC algorithm with cosine similarity for a speaker diarization task on CALLHOME dataset. In addition, the proposed approach also improves over the state-of-the-art system with PLDA affinity matrix with 10% relative improvement in DER.

Index Terms: Speaker Diarization, Self-supervised learning, AHC, Triplet loss, cosine similarity

1. Introduction

Speaker diarization, the task of determining who spoke when in a continuous multi-speaker audio stream, has received renewed interest in the recent years owing to the DIHARD evaluations [1] as well as the need for rich transcriptions in speech recognition systems [2]. The main challenges to speaker diarization include short speaker turns, noise/channel distortions and overlapping speech [3].

The prominent approaches to speaker diarization in the last decade have shifted to the use of embeddings from short windows of the incoming audio followed by a clustering process. Recently, the i-vector representations [4] have been replaced with neural embeddings from a time delay neural network, called x-vectors [5]. The neural embeddings have the advantage of utilizing large corpora of speaker supervised data for learning the background models [6]. While the embeddings used in diarization have advanced, the clustering approaches in many state-of-art systems use the traditional agglomerative hierarchical clustering (AHC) [7]. The improvements in the AHC algorithm targeted towards speaker diarization include pre-processing methods applied on embeddings like length normalization [8], recording level principal component analysis (PCA) [9] as well as the use of probabilistic linear discriminant analysis (PLDA) based affinity matrix computation for AHC [4]. However, none of these methods involve any form of learning in the clustering process.

In this paper, we propose a joint model for clustering and representation learning using the principles of self-supervised learning. Self-supervised learning is a branch of unsupervised learning where the targets for supervision are derived from the data itself [10, 11]. For speech processing, self-supervised learning has been mostly attempted for phoneme and speech recognition tasks [12]. In this paper, we explore self-supervised learning in the AHC framework where the steps of representation learning using AHC based self-supervision and the clustering of the representations are performed in an iterative fashion. With several experiments on LDC CALLHOME dataset, we illustrate the effectiveness of the proposed approach.

2. Related Prior Work

Most of the speaker diarization efforts with embeddings use a clustering based algorithm for diarization. The early approach using cosine distance with K-means or spectral clustering [13] has been advanced with PLDA based distance metric [4]. Narayanaswamy et. al. also proposed a metric learning paradigm for speaker diarization [14]. A joint learning paradigm for learning embeddings from speech activity detection, overlap detection and speaker discrimination has also been attempted by Filho et. al [15]. One of the drawbacks of the embedding based approach is the segmentation of the audio into fixed duration windows. Thus, speaker boundaries are constrained by the length of the windows. In order to overcome this, a re-segmentation framework using hidden Markov model and variational Bayes principles has also been explored [16, 17]. The clustering process may impair the model optimization aimed at minimizing diarization errors. To alleviate this problem, Zhang et al. [18] proposed a clustering-free diarization method. However, this method suffers from the requirements of large amounts of supervised multi-speaker conversational data to train the model.

In this paper, we propose a framework for joint learning of embeddings along with the clustering framework. The proposed work is inspired by Yang et. al [19], where joint unsupervised representation learning and clustering of images was explored. In our proposed approach, during training, clusters and representations are updated jointly. The clustering is a forward operation while the representation learning is a backward operation. The motivation for this framework is that good representations are beneficial to clustering and clustering results provide self-supervisory targets for representation learning. With the joint learning of the two processes in a single model, we can potentially obtain good representations of the audio data and precise speaker clusters.

This work was funded by British Telecom India Research Center (BTIRC) project on Speech Analytics.

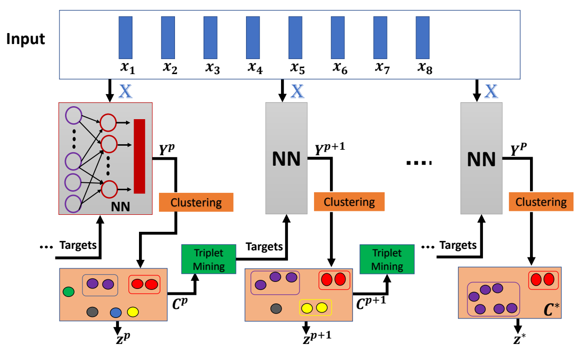

Figure 1: *Block diagram of the proposed deep self-supervised clustering algorithm.*

3. Proposed Approach

The proposed work is inspired by [19]. The block schematic of the proposed algorithm is shown in Figure 1. The proposed algorithm iterates between learning of representations using the DNN model with cluster labels as targets and the clustering of the representations using a modified AHC algorithm. The inputs are the x-vector features [5] where each x-vector feature is extracted from a audio segment of 1.5s duration with half overlap. More details of the time delay neural network (TDNN) based x-vector extractor are provided in Section 4.1.

3.1. Notations

The model parameters of the representation learning neural network (NN) are denoted by $\boldsymbol{\theta}$. Let p denote the iteration index of the proposed algorithm. Further, let

- $\mathbf{X}_r = \{x_1, ..., x_{N_r}\} \in \mathcal{R}^D$ denote the sequence of N_r input x-vector features for the recording r.
- $\mathbf{z}^p = \{z_1^p, ..., z_{N_r}^p\}$ denote the cluster labels for this recording, where each element takes a discrete cluster index value.
- $\mathbf{Y}^p = \{y_1^p, ..., y_{N_r}^p\} \in \mathcal{R}^d$ denote the output representations from the DNN model.
- $\mathbf{C}^p = \{\mathcal{C}_1^p, ..., \mathcal{C}_{N^p}^p\}$ denote the cluster set where each cluster $\mathcal{C}_i^p = \{y_k^p | z_k^p = i, \forall k \in \{1, .., N_r\}\}$. Here, N^p denotes the number of clusters obtained at the p-th iterative step.
- $\mathbf{Q}_{\mathcal{C}_i^p}^{K_c}$ denote the set of K_c nearest neighbour clusters of $\mathcal{C}_i^p$.
- $\mathcal{A}$ denote the affinity (or similarity) measure between two clusters.
- N^* denote the target number of clusters in the algorithm.

3.2. Iterative process

The algorithm is given in Figure 2. Here, we describe one step of the iterative process. At the p-th step, the DNN model parameters $\boldsymbol{\theta}^p$ are trained with the inputs as x-vector features $\mathbf{X}$ with a triplet loss defined using the cluster label indices $\mathbf{z}^{p-1}$ of the previous iteration. The DNN model is trained using several steps of gradient descent with this triplet loss. More details about the triplet mining for DNN training is given in the next section. Once the DNN model is trained, the last layer embeddings $\boldsymbol{Y}^p$ are derived by forward passing the x-vector inputs $\boldsymbol{X}$. Using the cluster definitions $\boldsymbol{\mathcal{C}}^{p-1}$ containing N^{p-1} clusters, the initial set of clusters $\boldsymbol{\mathcal{C}}^p$ are formed with representations $\boldsymbol{Y}^p$ (i.e., initially $N^p = N^{p-1}$). Then, the merging of the clusters is performed based on the affinity measure between pairs

Algorithm

```
Variables:
    $\boldsymbol{X} = \{x_1, ..., x_{N_r}\} \epsilon R^D$: X-vectors sequence of recording $r$
    $\boldsymbol{Y} = \{y_1, ..., y_{N_r}\} \epsilon R^d$: lower dimensional representations
    $\mathbf{z} = \{z_1, ..., z_{N_r}\} \epsilon R$: segment labels
    $\boldsymbol{\theta}$: NN parameters
    $(\boldsymbol{Y^p}, \boldsymbol{z^p}, \boldsymbol{\theta^p})$: refer to variables at iteration p
    $N^p$: Number of clusters at iteration p
    $N^*$: target number of clusters
Initialize: $(p = 0)$
    $\boldsymbol{\theta}^0 \rightarrow$ (Whitening + length norm + PCA)
    $\boldsymbol{Y}^0 \rightarrow$ PCA outputs
    $\mathbf{z}^0 \rightarrow$ Initial AHC ($N^0$ clusters) labels
Steps:
  While continue
    1. $p = p + 1$
    2. Sample triplets based on $\mathbf{z}^{p-1}$
    3. $\boldsymbol{\theta}^{p-1} \xrightarrow{NN\ with\ triplet\ loss} \boldsymbol{\theta}^p$
    4. $\boldsymbol{X} \xrightarrow{\theta^p} \boldsymbol{Y}^p$
    5. $\boldsymbol{Y}^p \xrightarrow{AHC(N^p\ clusters)} \mathbf{z}^p = \{z_1^p, ..., z_{N_r}^p\}$
    6. If stop:
    7.     break
Termination: $\{y_1^p, ..., y_{N_r}^p\} \xrightarrow{AHC(N^*\ clusters)} \{z_1^*, ..., z_{N_r}^*\}$
                                                    (Diarization output)
```

Figure 2: *Algorithm for joint learning using clustering based triplet loss.*

of clusters. More specifically, in a standard AHC algorithm, if $\mathcal{C}_a^p, \mathcal{C}_b^p$ are chosen as the two clusters to merge, then they satisfy,

$$\{\mathcal{C}_a^p, \mathcal{C}_b^p\} = \underset{\mathcal{C}_i^p, \mathcal{C}_j^p \in \mathbf{C}^p, i \neq j}{\operatorname{argmax}} \mathcal{A}\left(\mathcal{C}_i^p, \mathcal{C}_j^p\right) \tag{1}$$

In a modified version of the AHC merging cost, we can also incorporate discriminative term as follows,

$$\{\mathcal{C}_a^p, \mathcal{C}_b^p\} = \underset{\mathcal{C}_i^p, \mathcal{C}_j^p \in \mathbf{C}^p, i \neq j}{\operatorname{argmax}} \mathcal{A}\left(\mathcal{C}_i^p, \mathcal{C}_j^p\right) - \lambda \sum_{k=1}^{K_c} \mathcal{A}\left(\mathcal{C}_{ij}^p, Q_{\mathcal{C}_{ij}^p}^{K_c}[k]\right) \tag{2}$$

where $\mathcal{C}_{ij}^p = \mathcal{C}_i^p \cup \mathcal{C}_j^p$ denotes the merge of clusters i and j. Here, $Q_{\mathcal{C}_{ij}^p}^{K_c}$ denotes nearest neighbour clusters of $\mathcal{C}_{ij}^p$.

While the modified cost function in Eq. (2) is more computationally expensive compared to the direct cost function in Eq. (1), the modified cost function allows the clusters to be chosen for merging that are also distinct from other clusters in the mix. The parameter λ is a hyper-parameter and it acts as regularization between the intra cluster affinity versus inter cluster affinity. We perform multiple merges using the criterion defined above (Eq. (1)) reducing the total number of clusters in p-th iteration to the preset number of clusters N^p. If $N^p = N^*$, then the algorithm is stopped. Else, the iteration index is updated and the steps described above for the p-th iteration are repeated.

3.3. Triplet loss for DNN Training

For the DNN training at iteration p, the corresponding cluster label z_n^p for each input x-vector x_n is used where $n = 1, .., N_r$. The DNN model is trained by optimizing the triplet loss function in the following manner. For every cluster $\mathcal{C}_i^p$, a pair of embeddings forming the positive pair $\{y_a, y_b\} \epsilon \mathcal{C}_i^p$ is selected. The negative pair in the triplet is mined using the random sampling strategy. Here, we sample one y_c(negative) randomly from any other cluster $\mathcal{C}_j^p$ where $j \neq i$. Following the triplet mining, the

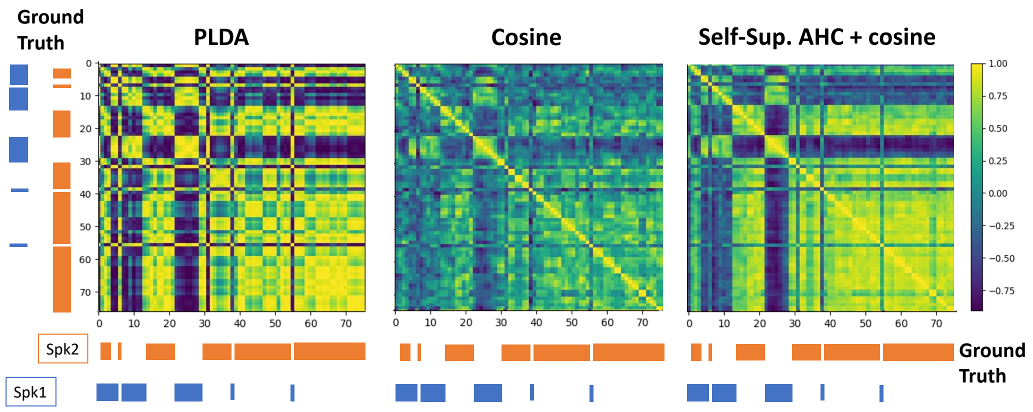

Figure 3: *Affinity matrices using PLDA, cosine and self-supervised AHC model with cosine for a 2 speaker recording from CH dataset. Ground truth labels are plotted across time in both sides of affinity matrices for comparison.*

model parameters of the DNN model are updated based on the following optimization criterion,

$$\boldsymbol{\theta}^p = \underset{\boldsymbol{\theta}}{\text{argmax}} \sum_{a,b,c} \left[\mathcal{A}(y_a, y_b) - \gamma \mathcal{A}(y_a, y_c)\right] \tag{3}$$

The above optimization is achieved using the iterative gradient descent algorithm. The hyper-parameter γ also controls the discriminability of the model learning.

3.4. Initialization

The DNN model used in our algorithm is a two layer feed-forward network. Motivated by the pre-processing steps in state-of-art diarization systems, the first layer of the DNN model is fed with x-vectors of D dimensions and performs a linear transform and a length normalization. The second layer of the DNN performs a dimensionality reduction to output representations of dimension d. The first layer of the DNN is initialized using a global whitening transform of the x-vectors computed using the training data. The second layer is initialized using a recording level principal component analysis (PCA) which reduces the D dimensional whitened x-vectors to d dimensions. The affinity measure used in this paper is based on cosine similarity measure. For initializing the clustering process, we use the AHC based clustering to N^0 clusters performed on the PCA components of the x-vectors.

3.5. Visualization of Affinity scores matrix

Figure 3 shows the PLDA and cosine affinity matrices for a 2 speaker recording from the CALLHOME dataset along with affinity matrix using cosine with self-supervised AHC. The figure shows that representations have become more discriminitive in the cosine space after self-supervised training. The affinity matrix is also more smoother compared to PLDA affinity matrix.

4. Experimental Setup

4.1. Baseline System

The baseline system for CALLHOME diarization uses the Kaldi recipe[1]. It involves extraction of 23 dimensional mel-frequency cepstral coefficients (MFCCs). A sliding mean normalization is applied over a 3s window. The speech segments (1.5s chunks with 0.75s overlap) are converted to 128 dimensional neural embeddings - x-vectors [6]. For the x-vector extraction, we use a time delay neural network (TDNN) with 5 layers followed by utterance pooling (with mean and standard deviation) and 2 affine layers mapping to the speaker targets. The output of first affine layer following the utterance level pooling is used as the x-vector embeddings [6].

The backend model is based on probabilistic linear discriminant analysis (PLDA) affinity score matrix. A Whitening transform trained on held out set is applied to the training set followed by length normalization [8]. An utterance level PCA is applied before PLDA scoring for dimensionality reduction [9]. The clustering is performed by AHC algorithm which hierarchically clusters the segments based on the PLDA affinity matrix and merges the clusters that represent the same speaker identity. The AHC stopping criterion is determined using heldout data.

4.2. Data

We evaluate our approach on CALLHOME (CH) dataset. The CH dataset is a collection of multi-lingual telephone data containing 500 recordings where the duration of each recording ranges from 2-5 mins per file. The number of speakers in each recording vary from 2 to 7, with majority of the files having 2 speakers. The CH dataset is equally divided into 2 different sets CH1 and CH2 with similar distribution of number of speakers. For training x-vector TDNN model, we used SRE 04-08, Switchboard cellular datasets as given in the Kaldi recipe. This training set had $4,285$ speakers.

4.3. Self-supervised AHC modeling

The experimental setup involves following steps:

- X-vector extraction: We use the same setup as in the baseline system to generate the x-vector features. The initialization of the DNN model uses whitening transform and the recording level PCA from the baseline system.
- DNN model: The first layer has 128 input and hidden nodes. The second layer has 10 output nodes (similar to the PCA dimension used in the baseline system). The learning rate is set as 0.001. The model is trained using the triplet loss defined in Eq. (3). The number of x-vectors per recording range from 50-700. Hence, we do not split the training triplet samples into minibatches, but rather perform a full batch training with Adam optimizer [20]. The stopping criterion is based on the parameter η which is the fraction of initial training triplet loss before the DNN training. The DNN training is stopped when η reaches a preset threshold (to avoid over fitting to

[1]https://kaldi-asr.org/models/m6

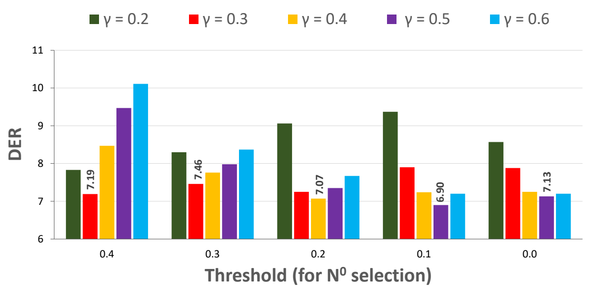

Figure 4: *Bar plot of DER vs threshold value used for selection of N^0 on CH1 subset for different choice of γ parameter with fixed $\lambda = 0.1$. The best DER results (known N^*) are obtained for $\gamma = 0.5$ with threshold=0.1.*

self-supervised cluster targets).

- Choice of hyper-parameters: We use K_c as 1 in all experiments. The value of λ is kept to 0 or 0.1. The parameter η is kept at 0.5. The parameter γ is varied between 0.2 and 0.5. Note that, $\lambda = 0.0$ is equivalent to the original AHC based triplet mining given in Eq. (1).
- Termination: To estimate the final number of clusters, we decide a threshold and stop the merging when the affinity scores between clusters are lower than the threshold. To find the optimum threshold, we find the DER over a range of thresholds and use the threshold which gives the best DER on the development set (similar to the baseline PLDA AHC system). For CH2 experiments, the CH1 dataset is used as the held-out set and vice versa.
- Performance metric : The performance metric used in our evaluation is Diarization Error Rate (DER) compute using a collar of 250ms and ignoring overlapping segments.

5. Results

We evaluate the system under two conditions - (i) using oracle number of speakers (known N^*), (ii) estimating the number of speakers based on held-out set (unknown N^*).

5.1. Choice of Initial Number of Clusters

Figure 4 shows the impact of number of initial clusters in the self-supervised AHC algorithm. Here, the results are reported for the case when the number of speakers is known (known N^*). We use the threshold value as stopping criteria for AHC to decide the initial number of clusters N^0 in the self- supervised training. As seen in the bar plot, x-axis shows the threshold values ranging from 0.0 to 0.4. Higher threshold signifies higher number of clusters as the merging will stop early. For each threshold, we plot DER of the self- supervised AHC model for different γ values from 0.2 to 0.6 and fixed λ = 0.1. As we increase γ, th DER decreases till certain value and then increases again. Also, as we reduce threshold, the N^0 moves closer to N^* and there is a significant improvement with higher γ. It is interesting to note that higher γ results in lower DER when the number of initial clusters reduces. This may be attributed to the fact that the higher value of γ makes the representations more discriminative across the clusters and this may result in aggressive merging of clusters thereby improving the DER. However, reducing number of clusters further starts degrading the purity of the cluster which also adversely impacts the representation learning model as well.

Table 1: *Comparison of Full CH dataset DER of different systems with baseline for both cases (i) known N^* and (ii) unknown N^**

System	λ	DER known N^*	DER unk. N^*
PLDA (Baseline)	-	**7.01**	**8.00**
Cosine	-	8.86	10.00
Self-supervised AHC	0.0	6.35	**8.26**
$[\gamma = 0.4, th = 0.1]$	0.1	6.47	8.33
(SSA1)	0.2	6.32	8.36
Self-supervised AHC	0.0	6.37	8.51
$[\gamma = 0.5, th = 0.1]$	0.1	**6.30**	8.60
(SSA2)	0.2	6.65	8.77
SSA1 + Baseline	0.0	**5.88**	**7.38**
SSA2 + Baseline	0.0	6.10	7.50

5.2. Full CALLHOME diarization Results

Table 1 shows the performance of the self-supervised AHC algorithm (SSA) compared to baseline using the known number of speakers (known N^*) as well as case when the number of speakers is chosen by the algorithm (unknown N^*). The table shows DER for different choice of λ from 0.0 to 0.2 and γ value of 0.4 and 0.5. The threshold value $th = 0.1$ is chosen to generate initial number of clusters N^0. In the current work, we run only two iterative steps of representation learning and clustering.

From Table 1, it can be observed that higher λ helps in improving DER when the number of speakers is known (known N^*). The best DER obtained for the proposed model (SSA2) is 6.30% where $\gamma = 0.5$ and $\lambda = 0.1$. The improvement of the self-supervised AHC learning over the cosine similarity baseline is 29%. The SSA system result is also relatively better than the PLDA baseline system by about 10%. These results show that SSA algorithm provides the advantage of learning the representations in the clustering process for improved diarization.

In the case with unknown N^* (number of speakers in the recording not known apriori), we find that the SSA algorithm is slightly inferior to the PLDA baseline although the model improves significantly over the cosine baseline. We have also performed fusion of SSA with a PLDA system by averaging the affinity matrix from the final SSA algorithm and the PLDA affinity matrix. Using this combined affinity matrix for AHC improves the DER significantly for both the known N^* case as well as the unknown N^* case. Overall, the fusion system (SSA+PLDA) relatively improves the PLDA baseline by 16 % on the known N^* case and about 8 % in the unknown N^* case.

6. Summary

In this paper, we have proposed an approach to jointly learn deep representations and speaker clusters to perform diarization. This algorithm is based on principles of self-supervised learning where the self-supervision is derived from the cluster identities provided by the AHC algorithm and is implemented at the recording level. Using an iterative procedure of representation learning and clustering, we show that the proposed self supervised AHC algorithm provides improved representations and precise speaker clusters. When the number of speakers in the given recording is known, the proposed algorithm improves the baseline AHC significantly. The method provides complimentary information to PLDA based AHC method and the fusion of the proposed approach with PLDA based diarization system improves the DER noticeably.

7. References

[1] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du, and S. Ganapathy, "The second DIHARD diarization challenge: Dataset, task, and baselines," in *Proc. of Interspeech*, 2019, pp. 978–982.

[2] S. Settle, J. Le Roux, T. Hori, S. Watanabe, and J. R. Hershey, "End-to-end multi-speaker speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4819–4823.

[3] M. H. Moattar and M. M. Homayounpour, "A review on speaker diarization systems and approaches," *Speech Communication*, vol. 54, no. 10, pp. 1065–1103, 2012.

[4] G. Sell and D. Garcia-Romero, "Speaker diarization with plda i-vector scoring and unsupervised calibration," in *IEEE Spoken Language Technology Workshop (SLT)*, 2014, pp. 413–417.

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

[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)*, 2018, pp. 5329–5333.

[7] W. H. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," *Journal of classification*, vol. 1, no. 1, pp. 7–24, 1984.

[8] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Twelfth annual conference of the international speech communication association*, 2011.

[9] W. Zhu and J. Pelecanos, "Online speaker diarization using adapted i-vector transforms," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5045–5049.

[10] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2051–2060.

[11] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *ICLR*, 2018.

[12] S. Pascual, M. Ravanelli, J. Serrà, A. Bonafonte, and Y. Bengio, "Learning problem-agnostic speech representations from multiple self-supervised tasks," *arXiv preprint arXiv:1904.03416*, 2019.

[13] S. Shum, N. Dehak, E. Chuangsuwanich, D. Reynolds, and J. Glass, "Exploiting intra-conversation variability for speaker diarization," in *Proc. of Interspeech*, 2011, pp. 945–948.

[14] V. S. Narayanaswamy, J. J. Thiagarajan, H. Song, and A. Spanias, "Designing an effective metric learning pipeline for speaker diarization," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5806–5810.

[15] V. A. Miasato Filho, D. A. Silva, and L. G. Depra Cuozzo, "Joint discriminative embedding learning, speech activity and overlap detection for the dihard speaker diarization challenge," in *Proc. of Interspeech*, 2018, pp. 2818–2822.

[16] M. Diez, L. Burget, and P. Matejka, "Speaker diarization based on bayesian hmm with eigenvoice priors." in *Odyssey*, 2018, pp. 147–154.

[17] P. Singh, H. Vardhan, S. Ganapathy, and A. Kanagasundaram, "LEAP diarization system for the second dihard challenge," in *Proc. of Interspeech*, 2019, pp. 983–987.

[18] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, and C. Wang, "Fully supervised speaker diarization," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6301–6305.

[19] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5147–5156.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.