The Second DIHARD challenge: System Description for USC-SAIL Team

Tae Jin Park, Manoj Kumar, Nikolaos Flemotomos, Monisankha Pal, Raghuveer Peri, Rimita Lahiri, Panayiotis Georgiou and Shrikanth Narayanan

University of Southern California, Los Angeles, CA, USA

{taejinpa, prabakar, flemotom, mp_323, rperi, rlahiri}@usc.edu,
{georgiou, shri}@sipi.usc.edu

Abstract

In this paper, we describe components that form a part of USC-SAIL team's submissions to Track 1 and Track 2 of the second DIHARD speaker diarization challenge. We describe each module in our speaker diarization pipeline and explain the rationale behind our choice of algorithms for each module, while comparing the Diarization Error Rate (DER) against different module combinations. We propose a clustering scheme based on spectral clustering that yields competitive performance. Moreover, we introduce an overlap detection scheme and a re-segmentation system for speaker diarization and investigate their performances using controlled and in-the-wild conditions. In addition, we describe the additional components that will be integrated to our speaker diarization system. To pursue the best performance, we compare our system with the state-of-the-art methods that are presented in the previous challenge and literature. We include preliminary results of our speaker diarization system on the evaluation data from the second DIHARD challenge.

Index Terms: speaker diarization, spectral clustering

1. Introduction

Speaker diarization has been considered as one of the most challenging tasks in the field of speech signal processing since it shows a degraded performance for conversations with frequent speaker turns and harsh acoustic conditions. The second DIHARD challenge dataset demonstrates these challenges well by including audio clips with a variety of speaker numbers or various acoustic conditions. Thus, we propose a range of methods to overcome these challenging problems. Firstly, we recognize speaker diarization for overlap speech and short segments are the most challenging task that cannot be covered by conventional segmentation-and-clustering algorithm. Thus, we employ overlap detection and re-segmentation components to tackle this issue. Secondly, there has been a need for speaker representation and clustering scheme robust to a slew of different acoustic conditions. Since the second DIHARD challenge provides a development dataset in diverse and challenging acoustic conditions, we use the same to find the best performing speaker representations and clustering methods. Our proposed speaker clustering scheme is compared with Agglomerative Hierarchical Clustering (AHC) algorithm that has been known to be state-of-the-art method. Moreover, we show a performance comparison between Probabilistic Linear Discriminant Analysis (PLDA) coupled with AHC and spectral clustering coupled with cosine similarity.

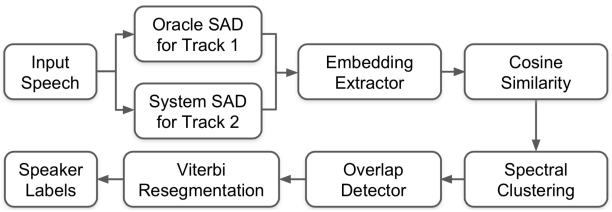

Figure 1: *The data flow of our proposed diarization pipeline*

2. Proposed Speaker Diarization System

The overall data flow of our best performing system is described in Fig.1. However, in the current section we describe all the methods we have experimented with and compare the performance with various combinations of components.

2.1. Data augmentation and feature extraction

Since DIHARD challenge targets a wide range of acoustic conditions and microphone setups, we curate our training data from multiple sources. We collect the train and test portions from NIST SRE (Speaker Recognition Evaluation) '04, '05, '06 and '08 and Switchboard corpora representing telephonic and broadcast news at 8kHz, and *dev* sets from Voxceleb 1 and 2 representing 'in-the-wild' speech at 16kHz. The combined data consists of 200,000 utterances spanning 8200 hours of speech. We perform three types of data augmentations: additive noise consisting of isotropic noises from the DEMAND [1] and REVERB [2] challenge database; reverberating the clean signal with room impulse responses from the RWCP sound scene database [3], the REVERB challenge database and the Aachen impulse response database (AIR) [4]; and finally reverberating the noisy signal. We opt to upsample all the audio files to 16kHz before the feature extraction process [5].

2.2. Speech Activity Detection (SAD)

Track 2 of DIHARD challenge requires diarization from scratch; hence the system needs an SAD module for detecting the speech and non-speech regions. A 3 layer fully-connected feedforward network is used for training the SAD. The training data comprises of 20 dimensional MFCCs computed from a subset of the augmented data mentioned in Section 2.1. The inputs are fed into the network with rectified linear unit (ReLU) activations. Each layer of the network contains 1024 neurons. The network is trained using *adam* optimizer to minimize the cross-entropy loss between predicted output labels and true output labels. Batch-normalization and dropout (*dropout rate = 0.3*) are used to the layers for regularization. During testing, Gaussian smoothing filter is employed on the posterior probabilities followed by thresholding for refining the SAD output.

Based on initial experiments, the threshold and standard deviation parameters for Gaussian smoothing filter have been chosen to be 0.85 and 1.4 respectively. The SAD system resulted in f1-score of 85.19% on DIHARD *dev* set.

2.3. Speaker Representation

2.3.1. X-vectors

X-vectors are fixed-dimensional speaker embeddings, known to achieve state-of-the-art performance for speaker recognition [6] and diarization [5] tasks. The first few layers of the neural architecture operate at the frame level and are inspired from the Time-Delay Neural networks (TDNNs) [7] where each layer sees a different temporal context. Then, a *statistics pooling* layer is used to collect the outputs of the last layer of the TDNN and compute the mean and standard deviation vectors. The next few dense layers operate at the segment level before a softmax inference layer maps segments to speaker labels. The activations of the first dense layer are selected as speaker embeddings. We use two different types of x-vector models. The first model, which we refer to as x-vec I, is an x-vector extractor[1] pre-trained on the Voxceleb 1 and 2 datasets following the neural architecture proposed in [6]. The second model, which we refer to as x-vec II, is also an x-vector extractor[2] trained on SRE data and SWBD datasets.

2.3.2. Hybrid DNN-TVM model

We employ a speaker embedding extractor based on the Hybrid DNN-TVM model that is introduced by Travadi *et. al.* [8]. Hybrid DNN-TVM (HDT) model incorporates TDNN in the initial layers and replaces the statistical pooling layer with the Total Variability Model (TVM). The output of TVM model is passed through an affine transformation and the transformed vector is sent to a fully-connected network. The entire network is then trained with cross-entropy loss. Thus, HDT model leverages the strength of both TDNN and TVM since TVM gives better fixed-dimensional representation of the TDNN output compared with the relatively simple mean and variance operations.

2.4. Denoising System

Motivated by the success of conditional generative adversarial network (CGAN) in speech enhancement [9] and i-vector transformation in short-utterance speaker verification [10], we propose to use CGAN on x-vector embeddings to compensate for additive noise in the speaker diarization framework. The approach is to train CGAN using both clean and noisy x-vectors, which can generate denoised x-vectors from noisy input x-vectors. For training the CGAN-based denoising model, we extract both the noise- and reverb- augmented x-vectors from the data generated by our augmentation process (Section 2.1), and corresponding clean x-vectors as the clean samples. In CGAN setup, both the generator (G) and discriminator (D) have a conditional input, which is the noisy x-vectors for our case. The generator tries to produce denoised x-vectors closer to clean x-vectors, while the discriminator tries to discriminate between clean and generator produced x-vectors. We use Wasserstein GAN [11] model in the CGAN framework. Furthermore, similar to [10], we incorporate multi-task training of G, where the generator network is integrated with another network $\mathrm{G_{sup}}$ for speaker prediction. The first section G is optimized to simultaneously reduce the generator loss, mean square error (MSE) loss and cross-entropy (CE) loss. MSE is computed between generator-produced x-vectors and clean x-vectors while training. The objective function of the second section, i.e., $\mathrm{G_{sup}}$ is to minimize CE loss. The output of the first section is directly fed to the second section. We expect a denoised output from the first section in addition to retaining speaker discriminative information due to the second section. After the completion of CGAN training, only the G network is used to produce a denoised x-vector. The initial ideas are explored in this paper, and more detailed models and experiments are planned for future work.

[1] http://kaldi-asr.org/models/m7

[2] http://kaldi-asr.org/models/m6

2.5. Distance measure

2.5.1. PLDA

Probabilistic Linear Discriminant Analysis (PLDA) [12, 13] provides a framework in which each data point is considered to be the output of a model which incorporates both within-individual and between-individual variation. Removing the channel-specific information, as proposed in [14], each speaker embedding $\boldsymbol{v}_i$ can be decomposed as

$$\boldsymbol{v}_i = \boldsymbol{m} + \boldsymbol{\Phi}\boldsymbol{\beta}_i + \boldsymbol{e}_i, \quad \boldsymbol{\beta}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \boldsymbol{e}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \quad (1)$$

and the similarity score between $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ can be computed via the hypothesis test

$$\mathrm{PLDA}(\boldsymbol{v}_i, \boldsymbol{v}_j) = \log\left(\frac{p(\boldsymbol{v}_i, \boldsymbol{v}_j|\text{same spk})}{p(\boldsymbol{v}_i|\text{dif. spk})p(\boldsymbol{v}_j|\text{dif. spk})}\right) \quad (2)$$

Three PLDA systems are evaluated: (a) the pre-trained PLDA model which comes with the pre-trained x-vector extractor, (b) a model trained on a subset of 128K utterances from the clean portion of our data, (c) a model trained on a subset of 64K utterances from the clean portion of our data, plus 64K utterances from the augmented portion of the data. In both cases (b) and (c), PLDA is trained on length-normalized [15] x-vectors, and the DIHARD development set is used to find the suitable whitening and centering transformations.

2.5.2. Cosine similarity

We employ cosine similarity (CS) only for spectral clustering since it achieves superior performance compared with PLDA score. For two arbitrary speaker embedding vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, cosine similarity is calculated as below:

$$cos(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{||\boldsymbol{x}|| \cdot ||\boldsymbol{y}||} \quad (3)$$

2.6. Clustering

2.6.1. Agglomerative Hierarchical Clustering

Agglomerative Hierarchical Clustering (AHC) is a traditional clustering scheme used in diarization systems [16]. Under this approach, clustering is done in an iterative manner where, at every step, two segments or sets of segments with the minimum distance are clustered together until a desired distance threshold is reached. The threshold is optimized on the development set. Instead of re-extracting the speaker embeddings at every step of the process, average linking is used [17].

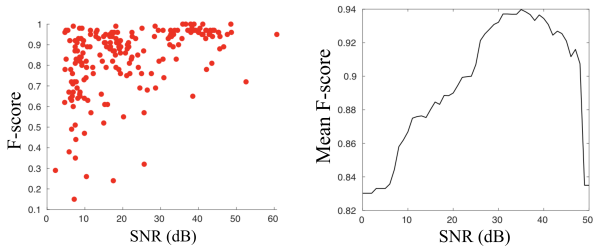

Figure 2: *(left) Overlap Detector performance vs SNR (right) Choosing the optimum minimum SNR*

2.6.2. *Spectral Clustering with Binary Affinity matrix*

We employ spectral clustering (SC) algorithm to cluster the speech segments using cosine similarity obtained from 2.5.2. Spectral clustering has been employed in a few studies [18, 19, 20] for speaker diarization. Unlike the pervious studies, we employ binary affinity matrix with pruning process that focuses only on the connectivity between segments. Moreover, we estimate the number of speakers before performing spectral clustering. The overall process of our spectral clustering scheme can be described as below:

1) **Input Matrix:** Given N segments, we obtain N by N affinity matrix $\mathbf{X}$ where each element is the cosine similarity between two segments. This input matrix is min-max normalized to 0-1 range.
2) **Binarization:** For each row, assign 0 if the cosine similarity is below the p-percentile in the row. If not, assign 1. We optimize the threshold ($p = 3$) on DIHARD *dev* set.
3) **Symmetrization:** To make undirected connections, we perform symmetrization as follows:

$$\mathbf{X_s} = \frac{1}{2}(\mathbf{X} + \mathbf{X^T}) \tag{4}$$

4) **Eigen Gap Analysis:** We perform eigen gap analysis to obtain the number of speakers. This helps to get accurate speaker labels while performing spectral clustering.
   1) Get Laplacian matrix by the following formula:

$$\begin{aligned} d_i &= \sum_{k=1}^{M} a_{ik} \\ \mathbf{D_c} &= \text{diag}\{d_1, d_2, ..., d_M\} \\ \mathbf{L_c} &= \mathbf{D_c} - \mathbf{X_s} \end{aligned} \tag{5}$$

   2) Perform Singular Value Decomposition (SVD) to obtain eigen values:

$$\mathbf{L_c} = \mathbf{U\Sigma V^T} \tag{6}$$

   3) Create an eigen gap vector as follows:

$$\mathbf{e_c} = [\lambda_2 - \lambda_1, \lambda_3 - \lambda_2, \cdots, \lambda_M - \lambda_{M-1}] \tag{7}$$

   4) Find the argument max from the vector $\mathbf{e_c}$

$$\widehat{n_s} = \min(\arg\max_n(\mathbf{e_c}), N_s) \tag{8}$$

   where we cap the maximum speaker number. We use $N_s = 8$ for our proposed system.
5) **Clustering on Spectral Domain:** From the eigen values we get from stage 4, we obtain $\widehat{n_s}$-eigen-vectors paired with $\widehat{n_s}$-smallest eigen values. For every segment, these eigen vectors are considered as $\widehat{n_s}$-dimensional spectral embedding. Spectral embeddings are then clustered by K-means clustering algorithm. We use the implementations of spectral clustering and K-means algorithm in [21].

Based on our experiments, our proposed spectral clustering scheme using cosine similarity results in lower DER when compared with AHC along with PLDA. There are two major factors influencing the performance gap between these two methods. First, we find that PLDA gives competitive performance when PLDA is trained and adapted on datasets which have acoustic conditions close to the test set. However, since the DIHARD evaluation dataset has large variability in acoustic conditions, PLDA faces significant mismatch between train and test conditions. Second, the optimal stopping criterion for AHC varies widely across different datasets and PLDA models. On the other hand, the best threshold value p for spectral clustering is relatively consistent over various datasets.

2.7. Overlap detection

We train a DNN based overlapped speech detector (OD) that classifies voiced speech into overlapped vs non-overlapped. We select AMI, ICSI (meeting corpora with significant overlapped speech) and CALLHOME (telephone speech) for training. We augment the training corpora with noise and reverberated speech, similar to Section 2.1. Spliced MFCC features (20-dimensional, $\pm$ 15 frames) are provided as input to the network. The neural network consists of 4 dense layers containing 256 neurons in each layer, followed by a statistics pooling layer and 2 embedding layers with 32 neurons each. The network is trained to minimize cross-entropy layer using *adam* [22]. Roughly 10 % of the train data is held-out to determine the optimum stopping criterion. On this held-out set, the system returned an f-score of 0.79. During evaluation on the *dev* portion of DIHARD data, we remove overlapped segments shorter than 0.2 seconds in order to remove false positives. This improved the mean f-score from 0.81 to 0.83.

We introduce overlapped speech to the output from clustering module as follows: At every frame labelled as overlapped speech (*current frame*), if a speaker change point occurs in the vicinity (0.25 seconds) we assign the new speaker as the second speaker for current frame. Speaker change point in this context is defined as change between speakers, and excludes changes between speakers and silence. We observed that this method did not result in overall DER change, but contributed to overall DER decrease after re-segmentation.

Considering that the DIHARD *dev* set consists of multiple domains with large acoustic and ambient noise variations, we suspected that the signal quality influences overlap detection performance. Using SNR as an estimate of signal quality, we observed that while sessions with high SNR resulted in better overlap detection performance, sessions with lower SNR exhibit wide variation (Figure 2). Hence, we choose an optimum SNR to maximize overlapped detector performance.

2.8. Viterbi Re-segmentation

The final component in our pipeline is Viterbi re-segmentation. Within each session, Gaussian mixture models with 128 components per speaker are trained from 20-dimensional MFCCs, with the number of components scaled down for speakers with too few frames. During decoding speaker posteriors are computed at every *block*, where a block is defined as 20 consecutive frames. Block decoding results in smoother speaker labels

Table 1: *Track 1: Reference SAD, DIHARD Dataset*

Systems	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
Embedding Extractor	x-vec I	x-vec I	x-vec I	x-vec I	HDT	x-vec I	x-vec I	x-vec I	x-vec I+II
Embedding Denoiser	-	-	-	-	-	ED	-	-	-
Distance Measure	PLDA(a)	PLDA(b)	PLDA(c)	CS	CS	CS	CS	CS	CS
Clustering Algorithm	AHC	AHC	AHC	AHC	SC	SC	SC	SC	SC
Overlap Detection	-	-	-	-	-	-	-	OD	-
Re-segmentation	-	-	-	-	-	-	-	Viterbi	-
DER: Dev Set	25.59	24.37	23.82	32.08	30.03	24.44	22.80	22.72	**21.82**
DER: Eval Set	25.44	25.99	25.75	32.78	30.85	27.59	24.21	24.22	**22.89**

Table 2: *Track 2: System SAD, DIHARD Dataset*

Systems	(9)	(10)
Embedding Extractor	x-vec I+II	x-vec I+II
Distance Measure	CS	CS
Clustering Algorithm	SC	SC
Overlap Detection	-	-
Resegmentation	-	Viterbi
DER: Dev Set	48.21	**47.79**
DER: Eval Set	46.78	**46.72**

and faster decoding time against frame-level decoding. The top two speakers (based on posterior probabilities) are assigned at frames with overlapped speech. Similar to overlap detector, the value of SNR is optimized to selectively apply re-segmentation.

3. Experimental Results

3.1. Track 1: Reference SAD

We compare the results from various combinations of components in Table 1. Systems (1), (2) and (3) test the performance of AHC algorithm with three different PLDA models. We can see that adapting the PLDA model to DIHARD *dev* set gives poor results on *eval* set since the PLDA c) model performs worse than the PLDA a) for *eval* set. Overall, the PLDA and AHC combination does not show a competitive performance compared with other combinations suggesting that using a very small subset of PLDA scores leads to a poor performance. We also test the performance of AHC coupled with cosine similarity in system (4). Note that system (4) shows very poor performance since AHC cannot be optimized with an accurate stopping criterion using cosine similarity. System (5) employs HDT embedding with spectral clustering and cosine similarity. We expect to gauge the contribution of speaker embedding by conducting the experiments with system (5). System (6) is tested with the embedding denoiser that is designed to mitigate the effect of noise. System (8) is our best performing system on *dev* set with x-vec I. The best performing system employs spectral clustering with cosine similarity. Since system (7) performs the best with x-vec I, we include overlap detection and Viterbi re-segmentation to system (7), resulting in system (8). Finally, in system (9), we show the performance with x-vec I+II which is a fusion of two different speaker embeddings. The fusion is done by adding two cosine similarity scores from two different x-vector extractors which are described in 2.3.1. We refer to this fusion approach as x-vec I+II as opposed to x-vec I.

3.2. Track 2: System SAD

We pick the best performing system (9) from Track 1 to evaluate the diarization performance with our proposed SAD system for Track 2. In addition, we also show the performance of system (10) from Track 1 to check the performance gain from the re-segmentation approach.

Table 3: *DER of system (8) on dev set with application of overlap detection and Viterbi re-segmentation. The overlap detector is selectively applied to sessions with high SNR.*

	Pre-Reseg.	**Post-Reseg.**	**Post-Reseg. (SNR)**
No Overlap	22.80	22.83	22.73
Overlap	22.79	22.75	22.72

4. Discussion

Our experiments on the second DIHARD challenge give us valuable lessons. First, we find that the biggest challenge is the mismatch between *dev* set and *eval* set. This problem is highlighted when PLDA and re-segmentation algorithm improve the performance on *dev* set while showing degradation on *eval* set. We plan to tackle this problem with more aggressive data augmentations and extensive grid searches on a wide selection of parameters. Second, we find that cosine similarity with spectral clustering gives competitive performance compared to PLDA model with AHC. The performance gain of cosine similarity with spectral clustering implies that PLDA model can suffer from the mismatch between *dev* set and *eval* set. Further investigation will be done on the comparison of distance measure to find out the best performing distance measure for challenging datasets. In addition, we will experiment with different ways to control the binarization threshold for spectral clustering. We find that SNR dependent processing improves the performance of overlap detection system and re-segmentation system since it mitigates the performance degradation for low SNR samples. The SNR dependent approach can be further extended to other components such as embedding extractor and distance measure. Finally, we find that the fusion of two different embeddings brings about a significant improvement on both DIHARD *dev* and *eval* set.

5. Conclusions

We showed how our proposed system improves the diarization performance against the baseline components showing performance gain from distance measure, clustering method, overlap detection and re-segmentaion. While further refinements are still required for our best performing modules, we will continue working on components that are not able to show performance improvements at the moment of writing this manuscript.

6. References

[1] J. Thiemann, N. Ito, and E. Vincent, "The diverse environments multi-channel acoustic noise database (demand): A database of multichannel environmental noise recordings," in *Proceedings of Meetings on Acoustics ICA2013*, vol. 19, no. 1. ASA, 2013, p. 035081.

[2] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, E. Habets, R. Haeb-Umbach, V. Leutnant, A. Sehr, W. Kellermann, R. Maas, S. Gannot, and B. Raj, "The reverb challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Oct 2013, pp. 1–4.

[3] S. Nakamura, K. Hiyane, F. Asano, T. Nishiura, and T. Yamada, "Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition," in *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC 2000, 31 May - June 2, 2000, Athens, Greece*. European Language Resources Association, 2000.

[4] M. Jeub, M. Schafer, and P. Vary, "A binaural room impulse response database for the evaluation of dereverberation algorithms," in *2009 16th International Conference on Digital Signal Processing*, July 2009, pp. 1–5.

[5] G. Sell, D. Snyder, A. McCree, D. Garcia-Romero, J. Villalba, M. Maciejewski, V. Manohar, N. Dehak, D. Povey, S. Watanabe *et al.*, "Diarization is hard: Some experiences and lessons learned for the jhu team in the inaugural dihard challenge," in *Proc. INTERSPEECH*, 2018, pp. 2808–2812.

[6] 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.

[7] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[8] R. Travadi and S. Narayanan, "Total variability layer in deep neural network embeddings for speaker verification," *IEEE Signal Processing Letters*, 2019.

[9] D. Michelsanti and Z.-H. Tan, "Conditional generative adversarial networks for speech enhancement and noise-robust speaker verification," *Proc. Interspeech 2017*, pp. 2008–2012, 2017.

[10] J. Zhang, N. Inoue, and K. Shinoda, "I-vector transformation using conditional generative adversarial networks for short utterance speaker verification," *Proc. Interspeech 2018*, pp. 3613–3617, 2018.

[11] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv preprint arXiv:1701.07875*, 2017.

[12] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision*. Springer, 2006, pp. 531–542.

[13] 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.

[14] P. Kenny, "Bayesian speaker verification with heavy-tailed priors." in *Odyssey*, vol. 14, 2010.

[15] 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.

[16] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A review of recent research," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 356–370, 2012.

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

[18] S. H. Shum, N. Dehak, R. Dehak, and J. R. Glass, "Unsupervised methods for speaker diarization: An integrated and iterative approach," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2015–2028, 2013.

[19] H. Ning, M. Liu, H. Tang, and T. S. Huang, "A spectral clustering approach to speaker diarization," in *Ninth International Conference on Spoken Language Processing*, 2006.

[20] Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno, "Speaker diarization with lstm," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5239–5243.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

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