Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 3138 – 3149

doi:10.3906/elk-1901-125

Unsupervised deep feature embeddings for speaker diarization

Rehan AHMAD*, **Syed ZUBAIR**
Department of Electrical Engineering, Faculty of Engineering and Technology, International Islamic University, Islamabad, Pakistan

Received: 18.01.2019 • **Accepted/Published Online:** 08.04.2019 • **Final Version:** 26.07.2019

Abstract: Speaker diarization aims to determine "who spoke when?" from multispeaker recording environments. In this paper, we propose to learn a set of high-level feature representations, referred to as feature embeddings, from an unsupervised deep architecture for speaker diarization. These sets of embeddings are learned through a deep autoencoder model when trained on mel-frequency cepstral coefficients (MFCCs) of input speech frames. Learned embeddings are then used in Gaussian mixture model based hierarchical clustering for diarization. The results show that these unsupervised embeddings are better compared to MFCCs in reducing the diarization error rate. Experiments conducted on the popular subset of the AMI meeting corpus consisting of 5.4 h of recordings show that the new embeddings decrease the average diarization error rate by 2.96%. However, for individual recordings, maximum improvement of 8.05% is acquired.

Key words: Diarization error rate, mel-frequency cepstral coefficients, hierarchical clustering, Gaussian mixture model, autoencoder

1. Introduction

Speaker diarization [1] is the process of partitioning an audio recording into speaker homogeneous regions. It answers the question of "who spoke when?" in a multispeaker environment. It is usually an unsupervised problem where the number of speakers and speaker-turn regions are unknown. The process automatically determines the speaker-specific segments and groups similar ones to form a speaker-specific diary. Its application lies in multimedia information retrieval, speaker recognition, and audio processing. Use cases of diarization include the analysis of speakers and their speech in meeting recordings, TV/talk shows, movies, phone conversations, conferences, or any other multispeaker recordings.

A typical diarization system consists of three main steps. The first one is a preprocessing step that consists of feature extraction such as mel-frequency cepstral coefficients (MFCCs), followed by a speech activity detection (SAD) step, which removes the silence and nonspeech regions from the speech, and finally a clustering and segmentation step, which works iteratively to segment speaker-change regions and collect homogeneous segments to make speaker-specific clusters. Typically, the hierarchical clustering [2] technique is initialized with a large number of clusters to iteratively merge similar clusters on the basis of the Bayesian information criterion (BIC) [3] to reduce the number of clusters to the actual number of speakers.

A noticeable improvement in diarization systems' performance was witnessed due the use of i-vectors [4–7], which model overall variability of speakers' voices and compress the information into low-dimensional space. With the rise in the use of neural networks and deep learning techniques, i-vector based methods

*Correspondence: rehan.ahmad@iiu.edu.pk

were outperformed by d-vector feature embeddings learned by neural networks [8]. Building upon the success of deep learning architectures, we propose feature embeddings learning based on autoencoders followed by hierarchical clustering for speaker diarization. In contrast to other deep learning based approaches, our method is unsupervised and directly matches the unsupervised nature of the speaker diarization system.

1.1. Related work

The earlier work on speaker diarization was started in 1998 [9], and then the series of National Institute of Standards and Technology (NIST) rich transcription (RT) evaluations were conducted from 2002 onward. In [10], the authors described agglomerative hierarchical clustering based on the HMM model with BIC measure to merge similar clusters. Their algorithm was tested on broadcast news. In [11], the authors further improved their algorithm by proposing an improved BIC measure and a purification module, which keep the clusters acoustically homogeneous in the clustering process. Furthermore, a beamforming technique was employed to enhance the signal quality for multiple distant microphones (MDMs). The improved technique was also tested on meeting recordings. Similarly, speaker diarization for RT'07 evaluation was described in [12], which employs a technique similar to that of [11] by keeping robustness and ease of portability. In this paper the authors focused on the improvement of speech activity detection to filter out audible nonspeech samples, which increased the accuracy of the system.

Recent developments in deep learning techniques created many opportunities in developing models that extract new sets of features either directly from raw datasets [13] or from hand-crafted features [14–16]. Recent work in speaker diarization consists of developing methods for the extraction of special features that help in speaker discrimination more robustly. For that purpose, many researchers have evaluated the feature embeddings technique using deep learning models such as long short-term memory (LSTM), deep neural networks (DNNs), and recurrent convolutional neural networks.

In [8], Wang et al. proposed the extraction of d-vectors [17] from the LSTM model for speaker diarization purposes. The model uses log-mel-filterbank frames as an input and uses the output frames of the LSTM architecture as a d-vector. It then applies diarization based on a spectral clustering algorithm. However, they applied diarization on speech frames that do not contain overlapping speech regions, which simplifies the diarization process. Moreover, LSTM was trained for the speaker verification task and the trained network was used to extract feature embeddings. Recently a fully supervised speaker diarization system was proposed by Zhang et al. [18] that utilizes an unbounded interleaved-state recurrent neural network (UIS-RNN) for diarization. Their system extracts speaker embedding (d-vectors) from the LSTM model and each speaker is modeled by a parameter-sharing RNN. The RNN model is further integrated with a distance-dependent Chinese restaurant process (ddCRP) to find the number of speakers in an audio recording. Cyrta [19] proposed a recurrent convolutional neural network (RCNN) to extract feature embeddings from magnitude spectrograms rather than from MFCC features. The RCNN based architecture was also trained for speaker classification tasks. Similarly, Romero et al. [20] proposed a DNN based feature embeddings technique. They replaced the conventional i-vector based feature with newly learned embeddings from the DNN architecture. The DNN architecture was specially designed based on network-in-network architecture (NIN). It was trained to jointly learn the discriminative embeddings and a scoring metric to measure the likelihood of segments generated from the same or different speakers. To train the architecture, training data were created by making pairs of the same and different speakers. Rouvier et al. [21] also proposed feature embeddings taken from the hidden layers of a DNN. The DNN architecture was trained to recognize speakers among 1000 speakers from a training set. This

trained architecture was then used to extract the new features. The i-vector based features were replaced by newly learned features for diarization. Sell et al. [22] described some experiences and lessons learned from the DIHARD diarization challenge. They described several key aspects of state-of-the-art diarization methods, such as feature extraction, feature embeddings (i-vector vs. x-vector), speech activity detection, and training data. Furthermore, the authors described their effective diarization system with wideband data, variational-Bayesian refinement, and single x-vector.

The above discussion shows that neural network based feature embeddings, known as d-vector embeddings, have improved speaker diarization performance as compared to i-vector based features. However, d-vectorbased embeddings were extracted from networks trained on speaker verification or classification tasks in supervised settings. Moreover, the pretrained networks used are trained on large datasets, which limits their use for relatively smaller datasets. We propose an unsupervised feature learning from a deep learning architecture, which closely resembles the original unsupervised speaker diarization pipeline.

1.2. Our contribution

Unlike the above discussed approaches, which used deep architecture with supervised training to get the feature embeddings, we propose a completely unsupervised technique to learn a new set of features from a deep network architecture. Our architecture is based on deep autoencoders, which consist of encoder and decoder parts. In our approach, the speech frames are converted into MFCC frames and five consecutive frames are grouped together to train the deep autoencoder such that the input and output are the same at training time. After the architecture has been trained, we use the encoder output as a new feature embedding for the diarization process. Moreover, we do not exclude the speakers' overlapping regions in the audio frames as was done in [8]. This makes the diarization process more difficult and usually increases the diarization error rate (DER). Furthermore, we have used a freely available subset of the AMI corpus consisting of a total of 5.4 h of recording. We compare our method with the Gaussian mixture model (GMM) based hierarchical diarization technique, which uses MFCC based speech features. Our method shows clear improvement over the baseline method in terms of DER.

The rest of the paper is organized as follows: Section 2 describes the methodology, which covers the preprocessing step, the baseline method for the diarization, and the proposed feature embedding technique. In Section 3 we describe the experimentation, which consists of a detailed description of the proposed architecture and its parameter settings. Moreover, this section also describes the evaluation metric for the diarization and the dataset used to test baseline and proposed methods. In Section 4, we present the results and a discussion. Finally, conclusions are drawn in Section 5.

2. Methodology

2.1. Conventional pipeline for speaker diarization

The conventional algorithm for speaker diarization consists of three steps. The first one is a preprocessing step that consists of extraction of MFCC features [23] by overlapping windows of speech samples. Typically, 13 or more MFCCs are used. The second step is speech activity detection (SAD) [12], which applies either an energy based measure to classify the speech and silence or a trained model based approach to classify speech/nonspeech audible sounds and silence. The SAD block eventually outputs the speech-only frames for further diarization. Accuracy of the subsequent segmentation and clustering step highly depends on the performance of SAD. The third step is the segmentation and clustering algorithm, which consists of a GMM/HMM based model with the

BIC [3] as a measure for matching similar clusters. An agglomerative hierarchical clustering based technique is used, which initializes a large number of GMM/HMM clusters, typically more than the number of available speakers, and trains those clusters on initially partitioned speech segments. Then it merges similar clusters based on the BIC metric and retrains the clusters based on resegmentation. The process of merging and retraining works iteratively until it is left with an optimal number of clusters. The final number of clusters eventually presents the number of speakers and their segments. Figure 1 shows the steps of a typical speaker diarization system.

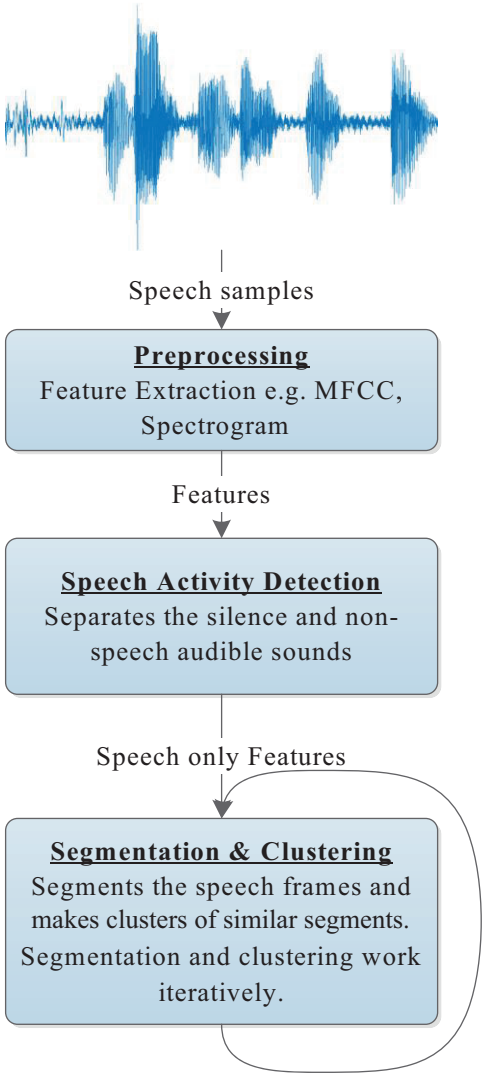

Figure 1. Steps for speaker diarization system.

2.2. Preprocessing

In the preprocessing step the audio recording is converted into the 19-dimensional MFCC features [23] and normalized by zero mean and unit variance. We use window lengths of 30 ms and hop-lengths of 10 ms in feature extraction. From the available annotations, we apply optimal SAD by setting nonspeech audio samples

to zero and apply an energy based SAD classifier. For this purpose, we train a support vector machine (SVM) classifier with the top 10% energy frames as speech and lowest 10% energy frames as nonspeech. The trained SVM is then used to classify the rest of the speech frames. This process significantly deceases the SAD error. Furthermore, after excluding the nonspeech MFCC frames, these basic features are used for diarization in the baseline method and used for feature embeddings in the proposed method.

2.3. State of the art

The baseline method for our comparison is the GMM based speaker diarization system discussed in [24]. We apply this method on mixed-headset audio recordings using MFCC features and compare it with our proposed feature embeddings. This method uses the GMM for segmentation and clustering with the agglomerative hierarchical clustering (AHC) technique, which iteratively merges similar clusters until the optimal number of clusters is obtained. The process initially partitions the whole audio recording into K segments and trains each cluster (GMM) on one segment. K is chosen to be much larger than the assumed number of speakers in the audio recording. A good guess for the initial number of clusters/segments for the audio recording of about 30 min is experimentally determined [9] as K = 16. Following are the diarization steps:

1. Partition the audio recording into K segments and train one GMM with each audio segment using the expectation-maximization (EM) algorithm.

2. Find the likelihood of each MFCC feature with each GMM. Then find the majority vote in 1.5 s of segment length and resegment the audio recording.

3. Retrain the GMMs with new segments using the same EM algorithm.

4. Find the BIC score of each GMM and then merge those GMMs that are most similar to each other. For this purpose, at each iteration, a new GMM is created and trained on the combined audio segments of those two GMMs that are to be matched, and then the difference in BIC scores of the new GMM and the merging candidate GMM is calculated. The improvement in BIC score merges the two GMMs permanently. The algorithm then goes to Step 2, such that it iteratively resegments the audio recording on the basis of majority votes and then merges the similar GMM clusters. GMM clusters are reduced iteratively until no two GMMs are the same. The algorithm then stops and presents the final number of clusters and segments. Finally, the diarization error rate is computed.

The applied BIC in diarization is defined as:

$$BIC = -log(p(D|\theta)) + L * 2 * log(N), \tag{1}$$

where $p(D|\theta)$ represents the likelihood of data D given the GMM model θ, L is the number of parameters of the model, and N represents the total number of speech frames in data D. The difference in BIC scores represented by ΔBIC for the combined GMM and the individuals is defined as follows:

$$\Delta BIC = log(p(D_m|\theta_m)) - log(p(D_a|\theta_a)) - log(p(D_b|\theta_b)), \tag{2}$$

where D_a and D_b are speech segments trained by GMM models θ_a and θ_b. D_m represents the combined data segments consisting of D_a and D_b, and θ_m represents the GMM model trained on D_m.

2.4. Feature embedding technique

In the proposed feature embedding method, we employ a deep autoencoder based unsupervised method to learn the new features from the basic features. Speaker diarization being an unsupervised problem, we propose the autoencoder based deep architecture to learn the features from unlabeled datasets. The autoencoder architecture has two parts: a feature encoding part, which is known as the encoder, and a feature decoding part, which is known as the decoder. A typical single-hidden-layer autoencoder can be represented as follows:

$$\mathbf{h} = a(W\mathbf{x} + b), \tag{3}$$

where $\mathbf{x}$ is an input vector for the autoencoder, W and b respectively represent the weight matrix and the bias of the encoder, a is a nonlinear activation function, and $\boldsymbol{h}$ represents the output of the encoder. The encoder output $\boldsymbol{h}$ is then fed into the decoder, which reconstructs the input to generate the output represented by $\hat{\mathbf{x}}$ as follows:

$$\hat{\mathbf{x}} = a(W^{'}\mathbf{h} + b^{'}), \tag{4}$$

where $W^{'}$ and $b^{'}$ represents the weight matrix and bias of the decoder, respectively, to reconstruct the original input. These equations can be extended for any large number of encoder and decoder layers. The proposed architecture consists of symmetric layers at the encoder and decoder sides.

Figure 2 shows the proposed architecture of the deep autoencoder. For such an unsupervised feature learning technique, we trained the architecture on the input data X with the same output labels X. The encoder part of the architecture provides the new set of features after the architecture has been trained. Shrinkage architecture [25] has been proposed to learn the low-dimensional features. To learn such low-dimensional features, initially we grouped five consecutive MFCC frames to create the input dimension of $19 \times 5 = 95$, and then we trained the architecture. The motivation is that speaker change usually does not occur in five successive frames and each speaker segment contains many successive MFCC frames.

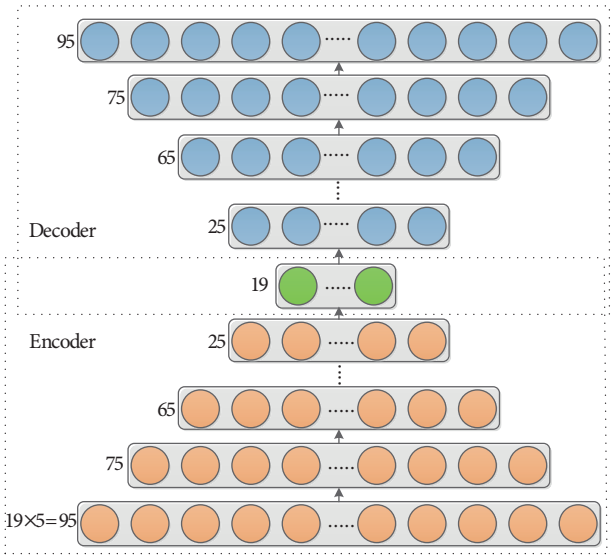

Figure 2. Proposed architecture of deep autoencoder.

The 95-dimensional basic features are converted into 19-dimensional features using the proposed architecture. These 19-dimensional features are further used in the diarization process. The advantage of using

shrinkage architecture [26] is that it reduces the model size, training time, and test time and does not compromise the accuracy of the system. Furthermore, this architecture also serves as a nonlinear PCA [27] for dimensionality reduction.

After we extract the new features from the encoder part of the trained architecture, we apply the unsupervised diarization process. This diarization process is the same as discussed in Section 2.3, which is based on segmentation and clustering using GMMs.

3. Experimental setup

The baseline method applies the GMM based segmentation and clustering on MFCC features. An agglomerative hierarchical clustering technique based on GMM is used with initially 16 clusters and 5 Gaussian mixtures in each cluster. The clusters are merged together based on the BIC and eventually it estimates the optimal number of clusters. For majority vote segmentation, the segment length of 1.5 s is considered. On AMI corpus recordings it was evaluated that segment lengths of 1.5 s show better results as compared to 2.5 s (which was suggested in the literature).

In the proposed architecture input and output are of 95 dimensions. The deep autoencoder is trained with a small batch size of 32 frames, 100 epochs, and Adadelta optimization. As labels of the architecture are the same as the training data, normalized to zero mean and unit variance, we used mean squared error (MSE) as an objective function. This objective function is more suitable and provides better convergence compared to the binary cross-entropy. It is represented as follows:

$$MSE = 1/N \sum_{i} (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2, \tag{5}$$

where $\mathbf{y}_i$ represents the input speech frame and $\hat{\mathbf{y}}_i$ represents the reconstructed speech frame by the autoencoder model. N represents the total number of speech frames. The MFCC features were normalized to zero mean and unit variance before applying feature embeddings, and so as to learn the new features more robustly we used the hyperbolic tangent (tanh) as an activation function in our deep model. After the model is trained, the encoder part is used to extract the 19-dimensional features as a representation of 95-dimensional input. The output of encoder is further normalized by zero mean and unit variance and then the same GMM based segmentation and clustering is applied as discussed for the baseline method. As 5 MFCC frames have already been grouped, we adjust the segment length in GMM training so that total segment length remains 1.5 s. For example, in the baseline method, taking 150 frames having hop lengths of 10 ms makes 1.5-s segments, and then in the proposed method as one feature is already a combination of 5 frames we used $150/5 = 30$ frames as a segment to make a length of 1.5 s.

3.1. Network architecture

The proposed feature embedding technique applies deep autoencoders with 13 hidden layers. The architecture is designed such that it shrinks at each layer. The input layer with 95 dimensions to the first hidden layer reduces the nodes by 20 and then each successive hidden layer has ten fewer nodes than the previous one, until it reaches the encoder output with 19 nodes. The architecture of encoder layers with number of nodes in each layer is represented as follows:

$$(input)\ 95 \rightarrow 75 \rightarrow 65 \rightarrow 55 \rightarrow 45 \rightarrow 35 \rightarrow 25 \rightarrow 19\ (encoder\ out). \tag{6}$$

Due to the symmetric architecture of the encoder and decoder, decoder layers have the same number of nodes at each layer as the encoder. The decoder architecture tries to reconstruct the encoded information from the encoder part. Nodes of the decoding layer are represented as follows:

$$19\ (encoder\ out) \rightarrow 25 \rightarrow 35 \rightarrow 45 \rightarrow 55 \rightarrow 65 \rightarrow 75 \rightarrow 95\ (out). \tag{7}$$

After the architecture has been trained, the encoder part provides the low-dimensional and newly learned features. To train the architecture, 5 MFCC frames were grouped together, thus making an input vector of 95 dimensions. This grouping was based on the assumption that speaker change does not usually occur frequently in five successive frames. It was experimentally determined that the group of 5 frames performs better than groups of 7 or more.

3.2. Computing environment

Our computing environment is based on Ubuntu Linux 17.10. The Librosa [28] package is used to extract the MFCC features in our Anaconda Python environment. Deep autoencoder architecture is designed in Keras, together with the Tensorflow [29] backend. To evaluate the diarization we compute the diarization error rate using the pyannote.metric [30] package. It includes computing four types of errors and aggregating them. These are speaker error, false alarm, confusion, and missed speech.

3.3. Evaluation metric

The diarization error rate (DER) [31] is used as an evaluation metric. It consists of computing speaker error, false alarm, missed speech, and confusion. It is defined as follows:

$$DER = E_{speaker} + E_{FalseAlarm} + E_{missed} + E_{confusion}. \tag{8}$$

The speaker error $E_{speaker}$ consists of segments assigned to the incorrect speaker. False alarm $E_{FalseAlarm}$ is defined as segments incorrectly assigned as speech and missed speech E_{missed} comprises the segments that have not been detected. Finally, confusion $E_{confusion}$ consists of overlapping speech assignments. Overlapping speech segments are also included in the computation of DER because this is usually part of audio recordings and plays a critical role in the performance of diarization systems.

3.4. Dataset

To evaluate our proposed method, we have used a popular subset of recordings from 12 meetings (5.4 h) from the publicly available AMI meeting corpus [32]. This corpus is available with a range of varying recording equipment, e.g., close talk microphones, lapels, and far-field and near-field microphone arrays. To compare our proposed method with the state of the art, we have used mixed-headset recording, which contains audio recordings in wav format. The selected subset of each audio recording contains four speakers. In this corpus, each meeting is recorded in four different sessions and independent recordings are provided. Each meeting has sessions of 15–35 min where each recording file has a meeting ID with small lettering that shows the session of that recording. For example, meeting IS1000a shows meeting ID 'IS1000' with 'a' being the recording of the first session and so on. The manual annotations of each session are also provided to check the validity of the diarization. Our method applies diarization on each recording independently and evaluates it in terms of DER.

Table 1. Average diarization error rate (%) of baseline method. The right-most column provides the average of each audio recording by running the algorithm 10 times. The last row presents the overall average of all the audio recordings.

Diarization error rate (DER) (%)											
Meeting ID	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Run 8	Run 9	Run 10	Average
IS1000a	36.85	42.21	41.72	42.19	42.35	41.93	42.09	42.17	46.95	42.33	42.079
IS1001a	41.62	41.48	41.9	42.61	42.07	41.76	42.00	42.74	42.52	42.74	42.144
IS1001b	48.88	48.34	48.71	48.7	47.57	48.67	48.16	47.6	48.71	47.67	48.301
IS1001c	52.3	53.08	53.68	52.08	52.99	53.89	53.11	52.4	52.27	53.09	52.889
IS1003b	50.44	51.14	50.27	50.73	50.54	50.17	48.64	50.71	51.15	63.02	51.681
IS1003d	68.38	68.81	68.61	67.5	69.00	67.87	69.04	68.83	68.57	67.82	68.443
IS1006b	59.74	49.74	50.04	66.32	49.63	42.63	49.61	59.48	49.76	49.55	52.65
IS1006d	66.02	66.95	66.89	67.07	66.90	66.67	66.96	67.13	67.03	66.87	66.849
IS1008a	11.87	20.25	20.38	20.53	11.89	11.87	11.96	12.20	20.34	20.43	16.172
IS1008b	12.08	12.02	12.73	11.85	11.91	11.74	12.54	11.79	11.87	12.22	12.075
IS1008c	41.16	40.46	39.81	40.91	39.64	41.19	40.52	40.71	39.98	41.52	40.59
IS1008d	38.23	37.61	26.21	38.00	37.55	38.1	37.85	37.63	26.23	37.62	35.503
Average DER for all the audio recordings											44.1146

Table 2. Average diarization error rate (%) of proposed feature embedding method. The right-most column provides the average of each audio recording by running the algorithm 10 times. The last row presents the overall average of all the audio recordings.

Diarization error rate (DER) (%)											
Meeting ID	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Run 8	Run 9	Run 10	Average
IS1000a	36.75	39.26	38.08	33.82	37.4	36.8	41.07	36.46	42.39	38.27	38.03
IS1001a	41.33	42.62	40.29	40.32	43.16	41.35	42.8	44.2	40.71	43.72	42.05
IS1001b	47.81	48.36	48.04	47.69	47.58	48.04	48.19	48.43	47.31	48.3	47.975
IS1001c	55.22	52.44	53.46	52.92	55.02	53.71	54.07	52.97	55.05	54.65	53.951
IS1003b	49.37	27.13	50.56	50.4	38.23	50.97	50.38	50.85	29.11	49.68	44.668
IS1003d	58.99	59.54	60.71	59.06	60.58	69.06	69.5	69.52	60.2	59.79	62.695
IS1006b	66.21	42.59	32.96	57.77	44.95	32.18	43.77	50.44	42.2	32.88	44.595
IS1006d	66.95	66.89	65.25	65.41	66.84	66.8	65.78	65.79	66.77	6.57	66.305
IS1008a	11.41	11.06	11.45	12.61	13.43	11.55	10.97	19.96	13.93	11.71	12.808
IS1008b	17.46	12.43	13.19	16.41	13.74	16.27	16.45	13.87	13.03	13.03	14.588
IS1008c	40.79	40.26	41.17	25.27	25.93	40.85	40.61	41.41	40.31	40.85	37.745
IS1008d	37.91	25.91	27.05	26.42	25.83	25.93	38.04	25.33	26.45	25.57	28.444
Average DER for all the audio recordings											41.1545

Table 3. Average diarization error rate (%) of proposed feature embedding method. The right-most column provides the average of each audio recording by running the algorithm 10 times. The last row presents the overall average of all the audio recordings.

Comparison table of DER (%)			
Meeting ID	Average results of baseline	Average results of FE method	Difference (improvement)
IS1000a	42.079	38.03	4.049
IS1001a	42.144	42.05	0.094
IS1001b	48.301	47.975	0.326
IS1001c	52.889	53.951	–1.062
IS1003b	51.681	44.668	7.013
IS1003d	68.443	62.695	5.748
IS1006b	52.65	44.595	8.055
IS1006d	66.849	66.305	0.544
IS1008a	16.172	12.808	3.364
IS1008b	12.075	14.588	–2.513
IS1008c	40.59	37.745	2.845
IS1008d	35.503	28.444	7.059
Average DER	44.1146	41.1545	-
Average improvement (%)	2.96		-

4. Results

Table 1 shows the computed DER for the baseline method (https://github.com/egonina/pycasp) on mixed-headset audio recordings. Due to random initialization in the GMM based method, which usually ends up at different local minima, we run the experiment on each audio recording 10 times and compute the average DER. The right-most column represents the average DER of each audio recording. Finally, overall average DER for all the audio recordings is computed, which is represented in the last row. It shows that on this subset of 12 audio recordings (5.4 h) the average DER of the baseline method is 44.11%.

Similarly, Table 2 shows the DER of the proposed feature embeddings (FE) method. The proposed method's overall average DER is 41.15%, which gives improvement of 2.96% as compared to the baseline method. Table 3 provides a comparison of average DER for each audio recording and presents improvement in the DER for each audio recording. The maximum improvement among the individual recordings has been observed for IS1006b, which is 8.05%. Overall, it has been observed that there is reduction of DER for all the recordings except two, and overall improvement is very significant.

5. Conclusion

In this paper we proposed a method for unsupervised feature embeddings extraction for speaker diarization. For this purpose, we made a deep architecture of an autoencoder, where the output of the encoder was selected as feature embeddings. The experiments showed that those feature embeddings improved the DER of the speaker diarization system as compared to conventional MFCC features. In particular, acquired features significantly improved the accuracy by reducing the average DER to 2.9% for collective datasets with maximum improvement gained up to 8.05% for one of the tested audio recordings.

References

[1] Anguera X, Bozonnet S, Evans N, Fredouille C, Friedland G. Speaker diarization: a review of recent research. IEEE Transactions on Audio, Speech, and Language Processing 2010; 20 (2): 356-370.

[2] Han KJ, Narayanan SS. Agglomerative hierarchical speaker clustering using incremental Gaussian mixture cluster modeling. In: Interspeech; Brisbane, Australia; 2008. pp. 20–23.

[3] Chen S, Gopalakrishnan P. Speaker, environment and channel change detection and clustering via the bayesian information criterion. In: Proceedings of DARPA Broadcast News Transcription and Understanding Workshop; Lansdowne, VA, USA; 1998. pp. 127–132.

[4] Dehak N, Kenny PJ, Dehak R, Dumouchel P, Ouellet P. Front-end factor analysis for speaker verification. IEEE Transactions on Audio, Speech, and Language Processing 2011; 19 (4): 788–798. doi: 10.1109/TASL.2010.2064307

[5] Madikeri S, Himawan I, Motlicek P, Ferras M. Integrating online i-vector extractor with information bottleneck based speaker diarization system. In: Interspeech; Dresden, Germany; 2015. pp. 3105–3109.

[6] Xu Y, Mcloughlin I, Song Y. Improved i-vector representation for speaker diarization. Circuits, Systems and Signal Process 2016; 35 (9): 3393–3404.

[7] Sell G, Garcia-Romero D. Speaker diarization with PLDA i-vector scoring and unsupervised calibration. In: IEEE 2014 Spoken Language Technology Workshop; Tahoe, NV, USA; 2014. pp. 413–417.

[8] Wang Q, Downey C, Wan Li, Mansfield PA, Moreno IL. Speaker diarization with lstm. In: IEEE 2018 International Conference on Acoustics, Speech and Signal Process; Calgary, Canada; 2018. pp. 5239–5243.

[9] Solomonoff A, Mielke A, Schmidt M, Gish H. Clustering speakers by their voices. In: Proceedings of the IEEE 1998 International Conference on Acoustics, Speech and Signal Processing; Seattle, WA, USA; 1998. pp. 757–760.

[10] Wooters C, Fung J, Peskin B, Anguera X. Towards robust speaker segmentation: the ICSI-SRI fall 2004 diarization system. In: RT-04F Workshop; Palisades, NY, USA; 2004. pp. 23-28.

[11] Anguera X, Wooters C, Peskin B, Aguiló M. Robust speaker segmentation for meetings: the ICSI-SRI spring 2005 diarization system. In: Renals S, Bengio S (editors). Machine Learning for Multimodal Interaction. Lecture Notes in Computer Science. Vol. 3869. Berlin, Germany: Springer, 2005, pp. 402-414.

[12] Wooters C, Huijbregts M. The ICSI RT07s speaker diarization system. In: Stiefelhagen R, Bowers R, Fiscus J (editors). Multimodal Technologies for Perception of Humans. Lecture Notes in Computer Science. Vol. 4625. Berlin, Germany: Springer, 2007, pp. 509-519.

[13] Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T (editors). Computer Vision - ECCV 2014. Lecture Notes in Computer Science. Vol. 8689. Berlin, Germany: Springer, 2014, pp. 818-833.

[14] Jati A, Georgiou P. Speaker2Vec: Unsupervised learning and adaptation of a speaker manifold using deep neural networks with an evaluation on speaker segmentation. In: Interspeech; Stockholm, Sweden; 2017. pp. 3567–3571.

[15] Snyder D, Ghahremani P, Povey D, Garcia-Romero D, Carmiel Y et al. Deep neural network-based speaker embeddings for end-to-end speaker verification. In: IEEE 2016 Spoken Language Technology Workshop; San Diego, CA, USA; 2016. pp. 165–170.

[16] Dey S, Koshinaka T, Motlicek P, Madikeri S, Lausanne D. DNN based speaker embedding using content information for text-dependent speaker verification. In: IEEE 2018 International Conference on Acoustics, Speech and Signal Processing; Calgary, Canada; 2018. pp. 5344–5348.

[17] Wan L, Wang Q, Papir A, Moreno IL. Generalized end-to-end loss for speaker verification. In: IEEE 2018 International Conference on Acoustics, Speech and Signal Processing; Calgary, Canada; 2018. pp. 4879–4883.

[18] Zhang A, Wang Q, Zhu Z, Paisley J, Wang C. Fully supervised speaker diarization. arXiv: 1810.04719v7, 2018.

[19] Cyrta P, Trzciński T, Stokowiec W. Speaker diarization using deep recurrent convolutional neural networks for speaker embeddings. In: Borzemski L, Świątek J, Wilimowska Z (editors). Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology - ISAT 2017. Advances in Intelligent Systems and Computing, Vol. 655. Berlin, Germany: Springer, 2017, pp. 107-117.

[20] Garcia-Romero D, Snyder D, Sell G, Povey D, McCree A. Speaker diarization using deep neural network embeddings. In: IEEE 2017 International Conference on Acoustics, Speech and Signal Processing; New Orleans, LA, USA; 2017. pp. 4930–4934.

[21] Rouvier M, Bousquet PM, Favre B. Speaker diarization through speaker embeddings. In: 23rd European Signal Processing Conference; Nice, France; 2015. pp. 2082–2086.

[22] Sell G, Snyder D, McCree A, Romero DG, Villalba J et al. Diarization is hard: some experiences and lessons learned for the JHU team in the inaugural DIHARD challenge. In: Proceedings of Interspeech; Hyderabad, India; 2018. pp. 2808–2812.

[23] Molau S, Pitz M, Schluter R, Ney H. Computing mel-frequency cepstral coefficients on the power spectrum. In: IEEE 2001 International Conference on Acoustics, Speech, and Signal Processing; Salt Lake City, UT, USA; 2001. pp. 73–76.

[24] Gonina E, Friedland G, Cook H, Keutzer K. Fast speaker diarization using a high-level scripting language. In: IEEE 2011 Workshop on Automatic Speech Recognition & Understanding; Waikoloa, HI, USA; 2011. pp. 553–558.

[25] Xu Y, Huang Q, Wang W, Foster P, Sigtia S et al. Unsupervised feature learning based on deep models for environmental audio tagging. IEEE/ACM Transactions on Audio, Speech, and Language Processing 2017; 25 (6): 1230–1241.

[26] Zhang S, Bao Y, Zhou P, Jiang H, Dai L. Improving deep neural networks for LVCSR using dropout and shrinking structure. In: IEEE 2014 International Conference on Acoustics, Speech and Signal Processing; Florence, Italy; 2014. pp. 6849–6853.

[27] Salakhutdinov RR, Hinton GE. Reducing the dimensionality of data with neural networks. Science 2006; 313 (5786): 504–507. doi: 10.1126/science.1127647

[28] McFee B, Raffel C, Liang D, Ellis DPW, McVicar M et al. Librosa: Audio and music signal analysis in python. In: Proceedings of the 14th Python in Science Conference; Austin, TX, USA; 2015. pp. 18–25.

[29] Abadi M, Barham P, Chen J, Chen Z, Davis A et al. Tensorflow: A system for large-scale machine learning. In: Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation; Berkeley, CA, USA; 2016. pp. 265-283.

[30] Bredin H. pyannote.metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems. In: Proceedings of Interspeech; Stockholm, Sweden; 2017. pp. 3587-3591.

[31] Galibert O. Methodologies for the evaluation of speaker diarization and automatic speech recognition in the presence of overlapping speech. In: Interspeech; Lyon, France; 2013. pp. 1131–1134.

[32] Carletta J, Ashby S, Bourban S, Flynn M, Guillemot M et al. The AMI Meeting Corpus: a pre-announcement. In: Renals S, Bengio S (editors). Machine Learning for Multimodal Interaction. Lecture Notes in Computer Science. Vol. 3869. Berlin, Germany: Springer, pp. 28-39.