Kaseva, Tuomas; Rouhe, Aku; Kurimo, Mikko

Spherediar: An Effective Speaker Diarization System for Meeting Data

Published in:
2019 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019 - Proceedings

DOI:
10.1109/ASRU46091.2019.9003967

Published: 01/12/2019

Please cite the original version:
ABSTRACT

In this paper, we present SphereDiar, a speaker diarization system composed of three novel subsystems: the Sphere-Speaker (SS) neural network, designed for speaker embedding extraction, a segmentation method called Homogeneity Based Segmentation (HBS) and a clustering algorithm called Top Two Silhouettes (Top2S). The system is evaluated on a set of over 200 manually transcribed multiparty meetings. The evaluation reveals that the system can be further simplified by omitting the use of HBS. Furthermore, we illustrate that SphereDiar achieves state-of-the-art results with two different meeting data sets.

Index Terms: speaker diarization, speaker embeddings, segmentation, spherical K-means, silhouette coefficients

1. INTRODUCTION

Speaker diarization answers the question “who spoke and when” \[1\]. In this process, a given audio stream is segmented into speaker turns: time intervals in which one speaker is speaking. It is determined, which of the speaker turns have the same speaker, but the actual identity (e.g. name) of the speakers is not required. Speaker diarization is a necessary subtask in many different speech applications such as creation of speech corpora, speech translation and speech recognition \[1, 2, 3\].

Speaker diarization is made difficult by the immense variability in speakers and recording conditions, and the unpredictable and overlapping speaker turns of spontaneous discussion \[1, 4\]. For these reasons, speaker diarization is still far from solved. In this paper, our main contribution is to propose a novel speaker diarization system which we have made available online \[1\]. The system consists of three main components which operate on three main tasks in speaker diarization: speaker modeling, segmentation and clustering \[1\].

The objective of speaker modeling is to embed a given speech utterance in a space which is more suitable for speaker discrimination \[5\]. Traditionally, this transformation has been performed with either Gaussian Mixture model (GMM) or i-vectors \[6, 7\]. Recently, also deep learning methods, both metric learning based \[2, 8, 9, 10\] and classification based \[11, 12, 13\], have been investigated. These methods have focused on creating neural speaker embeddings which have been shown to outperform i-vectors on many occasions \[12, 13, 14\]. Furthermore, especially classification based methods have shown great promise also in face verification \[15, 16\].

Motivated by these works, we choose to apply deep learning in our speaker modeling approach. We develop a novel neural network which learns the speaker embeddings through speaker classification. In this process, the network forces the embeddings to be \(L^2\) normalized, or in other words, spherical.

In our experiments, we show that this relatively simple operation has a profound positive impact on the speaker diarization task. Consequently, we name the network SphereSpeaker (SS), and our system SphereDiar.

In speaker diarization, segmentation refers to the task in which audio stream is divided into partitions which can be assigned to a single dominant speaker \[1\]. This procedure consists of speaker change detection (SCD) and overlapping speech detection (OSD) \[1, 17\]. Whereas hypothesis testing has been the standard approach in the former \[1, 9\], Hidden Markov models accompanied with GMMs have been in the latter \[1, 17\]. However, just as in speaker modeling, deep learning has recently been very successful in both OSD and SCD \[5, 18, 19, 20\]. Nevertheless, a segmentation approach which combines both OSD and SCD into a single process has not been proposed, although the connection of OSD and SCD has been well documented in literature \[17\]. In this paper, we develop such an approach, which we call Homogeneity Based Segmentation (HBS), and investigate its importance for our speaker diarization system. HBS uses deep learning and transforms the segmentation into a binary classification task.

The most popular clustering approach in speaker diarization has been agglomerative hierarchical clustering (AHC) \[1, 4, 14, 21\]. In addition, approaches exploiting Integer Linear Programming (ILP) \[22\], Information Bottleneck (IB) \[23\] and supervised learning \[24\] have been proposed. In our approach, we choose a slightly different clustering method which is based on using spherical K-means algorithm. This algorithm is essentially the same as K-means but uses cosine similarity as a distance distance metric and has \(L^2\) normalized cluster centers \[25\]. The choice of the algorithm is based on our preliminary experiments for clustering the speaker embeddings created with SS. However, the algorithm requires the number of cluster centers as an input, which is typically
unknown. Hence, in our method, we create multiple spherical K-means clusterings with a different number of clusters and choose the best clustering based on an empirically found and unsupervised criteria. These criteria are based on using silhouette coefficients [26] which, along with spherical K-means were also found to be beneficial for the clustering process. We call this method Top Two Silhouettes.

We show that our system achieves state-of-the-art results with a challenging dataset consisting of meeting recordings. Furthermore, we illustrate that these results are obtained even without using HBS and that HBS has overall a little significance for our system. As a consequence, our system can then be simplified considerably by excluding segmentation entirely. This is not only convenient but also an interesting discovery since especially OSD has been a prominent research direction in speaker diarization [1, 4, 17, 20].

Since speaker turns change unpredictably in spontaneous discussion, each two-second frame can include speech from multiple different speakers. That is, in general, each speaker speaks for only some percentage of the frame’s duration. For each frame s, we compute a quantity we call homogeneity percentage \(H_{\%}\). It is the highest percentage of frame time covered by a single speaker. The frame’s speaker label \(l\) is this most prominent speaker. Equivalently,

\[
l = \arg \max_i |T_i|, \quad H_{\%} = \frac{\max_i |T_i| - 1}{|T|} \times 100\%, \tag{1}
\]

where \(T = \{T_{-1}, T_1, ..., T_{n_s}\}\) is a set of transcription labels of s with \(T_{-1}\) corresponding to samples which include overlapping speech and \(T_{i\neq-1}\) depicting samples which are assigned to a speaker \(i\).

### Table 1. Removed meetings.

<table>
<thead>
<tr>
<th>AMI</th>
<th>EN2001a, EN2001e, EN2002c, EN2003a, EN2006a, EN2006b, IB4005, IS1003b</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICSI</td>
<td>Bm012</td>
</tr>
</tbody>
</table>

The speaker corpora comprise of four different partitions, \(LS_{1000}\) and \(LS_{2000}\) which are collected from Librispeech corpus and \(VC_{1000}\) and \(VC_{2000}\) which are extracted from the Voxceleb2 dataset [3, 29]. The number of speakers in a partition is given as the subscript. To the best of our knowledge, the speakers in each partition are disjoint from the speakers in the meeting corpus. The speech material of each partition consists of frames of 2s duration which are extracted without overlap. The sampling frequency is the same as with the meeting corpus. In the extraction procedure, we use the webrtc speech activity detection (SAD) system [30], since reference transcriptions are not available. The gender distributions and the frame compositions are depicted in Table 2 and 3.

In order to balance the speaker label distributions with the partitions with the same number of speakers, the maximum number of frames per speaker is limited. The limit for the partitions \(LS_{2000}\) and \(VC_{2000}\) is assigned as 670 whereas the limit for \(LS_{1000}\) and \(VC_{1000}\) is 1000. The \(LS_{1000}\) partition, however, did not include quite as much speech material as \(VC_{1000}\), so the maximum number of frames per speaker is only 764.

### 3. SPHEREDIAR

The block diagram of SphereDiar speaker diarization system is presented in Figure 1. In this Figure, input \(S\) depicts a sequence of 2s frames sampled with 16 kHz frequency and the output \(L\) the corresponding speaker label sequence. Note that we do not provide SAD in this system. This is by no
Table 2. Gender distribution in speaker corpora.

<table>
<thead>
<tr>
<th></th>
<th>Number of females</th>
<th>Number of males</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LS</strong></td>
<td>1000</td>
<td>500</td>
<td>1000</td>
</tr>
<tr>
<td><strong>LS</strong></td>
<td>2000</td>
<td>987</td>
<td>1013</td>
</tr>
<tr>
<td><strong>VC</strong></td>
<td>1000</td>
<td>500</td>
<td>1000</td>
</tr>
<tr>
<td><strong>VC</strong></td>
<td>2000</td>
<td>731</td>
<td>1269</td>
</tr>
</tbody>
</table>

Table 3. Frame compositions in speaker corpora.

<table>
<thead>
<tr>
<th></th>
<th>Minimum number of frames per speaker</th>
<th>Maximum number of frames per speaker</th>
<th>Total number of frames</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LS</strong></td>
<td>1000</td>
<td>382</td>
<td>764</td>
</tr>
<tr>
<td></td>
<td></td>
<td>654 297</td>
<td></td>
</tr>
<tr>
<td><strong>LS</strong></td>
<td>2000</td>
<td>341</td>
<td>670</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 204 967</td>
<td></td>
</tr>
<tr>
<td><strong>VC</strong></td>
<td>1000</td>
<td>838</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>995 443</td>
<td></td>
</tr>
<tr>
<td><strong>VC</strong></td>
<td>2000</td>
<td>577</td>
<td>670</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 337 601</td>
<td></td>
</tr>
</tbody>
</table>

means a trivial exclusion since SAD is an essential component in any speaker diarization system [1]. However, when diarization systems are developed, reference SAD labels are often used in order to focus on the actual speaker diarization [17, 20, 21, 31]. This is also the case with the speaker diarization systems against which we compare our system in section 4.

Feature extraction. In the beginning of the diarization procedure, each frame s in S is converted to \( x \in \mathbb{R}^{201 \times 59} \), which consists of a sequence of 19 Mel-Frequency Cepstral Coefficients (MFCC), their first and second derivatives, and the first and second derivatives of energy just as in [32]. MFCCs are extracted every 10ms with a 25ms window duration using Librosa [33] and normalized with zero mean and unit variance.

SphereSpeaker. In speaker modeling, each feature sequence \( x \) is projected into a speaker embedding \( f(x) \). The projection is attained by using the neural network depicted in Figure 2 and Table 4. This network is initially designed to predict a class, or in our setting, a speaker identity for \( x \). Consequently, the final layer has a softmax activation function which assures that the output is an \( N_s \) dimensional probability distribution, where \( N_s \) is the number of classes. The speaker embedding \( f \) is produced in this classification process as the output of the last hidden layer. As a result, the final layer is only used during the training.

The network consists of two main components: a cascade of three bidirectional Long Short-Term Memory (LSTM) neural networks with skip connections which adheres to the architecture of [32] and an embedding layer. In this layer, we assign two conditions on the embedding: \( f \in \mathbb{R}^{1000} \) and \( \|f\|_2 = 1 \). The use of \( L^2 \) normalization is influenced by the work in [12, 16] whereas the embedding dimension and the overall configuration of the embedding layer are based on our preliminary experiments. The importance of the normalization operation inside the network will be emphasized further in the experiments section where we compare SS with SS*. The latter is otherwise the same network as SS, but does not include \( L^2 \) normalization layer. Instead, the speaker embeddings extracted with this network are \( L^2 \) normalized externally.

Table 4. Output dimensions of each layer in SphereSpeaker and HBS neural networks.

<table>
<thead>
<tr>
<th>SphereSpeaker neural network</th>
<th>Output dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bidirectional LSTM₁</td>
<td>201 × 500</td>
</tr>
<tr>
<td>Bidirectional LSTM₂</td>
<td>201 × 500</td>
</tr>
<tr>
<td>Bidirectional LSTM₃</td>
<td>201 × 500</td>
</tr>
<tr>
<td>Concatenation</td>
<td>201 × 1500</td>
</tr>
<tr>
<td>Embedding layer</td>
<td>1000</td>
</tr>
<tr>
<td>Fully connected layer (softmax)</td>
<td>( N_s )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HBS neural network</th>
<th>Output dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bidirectional LSTM</td>
<td>201 × 600</td>
</tr>
<tr>
<td>Attention layer</td>
<td>201 × 600</td>
</tr>
<tr>
<td>Average pooling layer</td>
<td>600</td>
</tr>
<tr>
<td>Fully connected layer (sigmoid)</td>
<td>1</td>
</tr>
</tbody>
</table>

Homogeneity Based Segmentation. Segmentation is performed as a binary classification where the formulation of classes is based on the concept of homogeneity percentage. In this approach, our aim is to label each \( x \) as 0, if \( H \% \) of the
Here, each frame’s of x exceeds a given threshold $H_{\theta \%}$ and otherwise as 1. As a result, we call this method Homogeneity Based Segmentation. Ideally, due to the definition of the homogeneity percentage, class 1 consists of frames which include speaker change boundaries and overlapping speech whereas class 0 comprises of frames which can be assigned to a single dominant speaker. Nevertheless, a gray area between classes does exist when homogeneity percentages are close to the threshold $H_{\theta \%}$. Moreover, there is no optimal threshold: we set $H_{\theta \%} = 65\%$ in our experiments as we consider it to be a suitable compromise. The ultimate goal of HBS is to exclude the frames assigned to class 1 from the clustering procedure. The main feature of the HBS is that in theory, it allows performing both OSD and SCD in a single process. Such simplification is yet to be proposed or experimented in speaker diarization.

The class labels are predicted with the neural network illustrated in Figure 3 and in Table 4. The two main components of this network are the bidirectional LSTM neural network which is motivated by the works in [9] [18] [19] and the attention layer which is based on the implementation in [34]. With the former, we use both regular and recurrent dropout and assign both dropouts as 0.2. All other layers are chosen based on our preliminary experiments. The class $h(x) \in \{0, 1\}$ of each x is determined based on rounding the output of the network $h(x) \in [0, 1]$ to the nearest integer.

![HBS neural network](image)

**Fig. 3.** HBS neural network.

**Top Two Silhouettes.** After the speaker modeling and segmentation we have obtained a sequence of speaker embeddings $F$ and a sequence of HBS labels $H$. As a final step, we assign each $f$ in $F$ with a speaker label. In our approach, this assignment is determined by clustering $E$, a subset of $F$ consisting of embeddings $f_i$ with the HBS label $h_i = 0$. The clustering is performed with a novel algorithm which can be divided into two steps: the proposal generation and the optimal proposal determination.

In the first step, $E$ is fitted with multiple different spherical K-means configurations with $K$ ranging from 2 to $N_{\text{max}}$. Here, $N_{\text{max}}$ refers to an initial guess of a maximum number of speakers in $E$. Each configuration is run with $R$ different initializations from which the final configuration is determined based on the run which yielded the highest silhouette score. This score is the average of silhouette coefficients which are computed for each speaker embedding. In this computation, cosine similarity is used as a distance metric. More details of the calculation of the coefficients can be found in [26]. The proposals $P_i$ are then created based on these final configurations.

In the second step, the optimal proposal $P_{\text{opt}}$ is chosen. First, the proposals corresponding to the two largest silhouette scores, $P_{\text{top-1}}$ and $P_{\text{top-2}}$ are recovered. If (i) $P_{\text{top-1}}$ has more clusters, or (ii) the silhouette score of $P_{\text{top-2}}$ is below a threshold $\delta$, then $P_{\text{opt}} = P_{\text{top-1}}$. This is a heuristic rule which we have found experimentally and can be interpreted as a further confidence that $P_{\text{top-1}}$ is the optimal proposal.

Otherwise, if both (i) and (ii) are unsatisfied, the algorithm deduces that $P_{\text{top-2}}$ could also be chosen. As $P_{\text{top-2}}$ has then more clusters than $P_{\text{top-1}}$, the algorithm investigates if any of the clusters in $P_{\text{top-1}}$ might contain inner clusters. This investigation is performed in a similar fashion as in the first step but for each each cluster in $P_{\text{top-1}}$. The assignment $P_{\text{opt}} = P_{\text{top-2}}$ is then chosen if for any initialization or cluster, both maximum silhouette value is above $\delta$ and a corresponding $K \in \{2, 3\}$. In this condition, the maximum number of inner clusters is restricted to 3 since a higher number would be highly improbable. However, if this condition is not satisfied the algorithm again chooses $P_{\text{opt}} = P_{\text{top-1}}$.

**Algorithm 1: Top Two Silhouettes**

**Input:** Set of speaker embeddings $E$, a number of initializations $R$, a maximum number of speakers $N_{\text{max}}$ and a threshold $\delta$

**Output:** Proposal $P = \{L, C\}$

**Steps:**

1. Initialize $K = \{2, \ldots, N_{\text{max}}\}$ and $s = \{0, \ldots, 0\}$, $|s| = |K|$

2. **for** $r = 1$ to $R$ **do**
   
   **for** $i = 1$ to $|K|$ **do**
   
   $\phi(K_i, E) \rightarrow L_i \rightarrow v(L_i, E) \rightarrow \hat{s}_i$
   
   if $(\hat{s}_i > s_i)$ $\rightarrow s_i = \hat{s}_i$.

3. Find largest and second largest silhouette scores $s_{\text{top-1}}$ and $s_{\text{top-2}}$, respectively.
   
   **if not** $(\text{top-2} > \text{top-1} \land s_{\text{top-2}} > \delta)$
   
   $\rightarrow$ **return** $L_{\text{top-1}}, C_{\text{top-1}}$

4. Repeat step 2 for each $E_i \in E = \{f_i \mid l_i = k \in L_{\text{top-1}}\}$ In the process, for any $j, r$
   
   **if** $(\max_j v(L_{ij}, E_j) > \delta \land K_i \in \{2, 3\})$
   
   $\rightarrow$ **return** $L_{\text{top-2}}, C_{\text{top-2}}$

5. **return** $L_{\text{top-1}}, C_{\text{top-1}}.$
The labels $L$ for $F$ are then generated using associated cluster centers $C_{opt}$ of $P_{opt}$. As the proposals corresponding to the two largest silhouette scores are central to the algorithm, we have named it Top Two Silhouettes. In the experiments section, we demonstrate the validity of this algorithm by comparing it with Top Silhouette (TopS), which is essentially the same as Top2S but always assigns $P_{opt} = P_{top-1}$.

Top Two Silhouettes is described more formally in Algorithm 1. In this description, spherical K-means is denoted with $\phi$ and the calculation of the silhouette score with a variable $v$. Moreover, instead of two steps, the description consists of three main steps consisting of the calculation of the silhouette scores, the evaluation of the conditions (i) and (ii) and the possible inner cluster search.

4. EXPERIMENTS

4.1. Experimental setup

**Evaluation metric.** All experiments are conducted using the same evaluation metric called diarization error rate (DER) [11]. In general, DER consists of SAD related errors (false alarm and false rejection) and speaker errors which, in our case, can be interpreted as a clustering errors between reference and predicted speaker labels [11]. However, since we have performed SAD on all meetings in the meeting corpus as a preprocessing step, the computation of DER simplifies to a calculation of the speaker error which we compute with Hungarian algorithm [35]. Furthermore, when calculating DER, we consider only labels corresponding to the frames which have $H_p$ above the threshold $H_{0\%} = 65\%$ unless explicitly mentioned otherwise.

**Neural network training and evaluation.** We train 10 models in total: 8 for speaker modeling and 2 to be used for segmentation. The first eight models are trained using SS and SS* and four different training and evaluation set splits. The splits are generated from each partition in the speaker corpora by choosing randomly 45 frames from each speaker for testing and leaving the rest for training.

The last two models both use HBS neural network, but are trained solely using the meeting corpus with two different evaluation sets: $AMI_{eval}$ which is a same as in [21] or $ICSI_{eval}$ consisting of 9 ICSI meetings [1]. In both cases, all other meetings in the meeting corpus are reserved for training. Moreover, only frames which have $H_{\%} = 100\%$ (labeled as 0) and frames with $H_{\%} \leq 65\%$ (labeled as 1) are used in training and evaluation. This choice is based on ensuring proper discrimination between classes that we found beneficial in our preliminary experiments.

All 10 models are trained using Keras deep learning library [36] with batch size 256, for 45 epochs. We use the cross entropy as a loss function and using Adam [37] optimizer. When training the last two models, we also weight class 1 twice as much as class 0 in order to balance the class distributions.

**Clustering parameters.** We assign $R = 50$ and $N_{max} = 11$ in all experiments. We choose to set $R$ this high since spherical K-means has a tendency to converge to a local maximum [25]. The value of $N_{max}$ is selected to exceed the highest possible participant number, 9, of the meetings in the meeting corpus. In addition, we set $\delta = 0.1$, which we attained by conducting a grid search on a clustering development set $Clust_{dev}$ of 12 meetings extracted from the meeting corpus. This set is disjoint with both $AMI_{eval}$ and $ICSI_{eval}$. In the grid search, we evaluated each threshold using DER, did not use HBS and performed speaker modeling with Sphere-Speaker trained with $VC_{1000}$.

4.2. Results

![Figure 4: Average DER over 225 meetings from the meeting corpus with different SphereDiar configurations which omit HBS.](image)

In Figure 4, we visualize speaker diarization results with 225 meetings from the meeting corpus that are disjoint with $Clust_{dev}$. These results are obtained using all possible SphereDiar configurations introduced in this paper but without using HBS ($h_i = 0, \forall i$) as most of the meetings have been used in HBS training. The results illustrate that SS outperforms SS*, especially when these neural networks are trained with Voxceleb2 partitions, and that Top2S performs markedly better than TopS. Moreover, the results show that both the increase in the number of training speakers and the use of Voxceleb2 partitions over Librispeech partitions are preferable in speaker modeling training. The best configuration is attained by combining SS trained with Librispeech partitions and Top2S and it achieves $3\%$ average DER over the 225 meetings.

The results in Table 5 show that our HBS system fails to benefit the speaker diarization task. In the experiments which were briefly discussed with neural network training and evaluation, the HBS system achieved mean average precision of 0.953 with $AMI_{eval}$ and 0.935 with $ICSI_{eval}$. Clearly, these scores were not high enough to make HBS beneficial for speaker diarization.
be interpreted as shrinking the collar around speaker change boundaries. This decrease allows the average DER comparison to be as fair as possible since any further decrease in the value of $H_{55\%}$ results in severe difficulties of labeling the frames accurately. Consider, for instance, the example given in subsection 2.1.5. If a frame would have $H_{55\%} = 50\%$, and would contain two speakers without any overlapping speech. Then, the speaker label of this frame could not be determined.

The results illustrate that our system is able to outperform the systems in [21, 38]. Our result is particularly better when comparing to [38] but we admit that our system has been trained with Voxceleb2 which was not available at the time for [38]. However, the system in [21] has been trained with a very similar data as ours, using Voxceleb [2] and other relevant datasets, but our result is still better. Moreover, as we do not use HBS in the comparison, our domain adaptation is only based on 12 meetings in Clust_dev. This is significantly less than used in either [21] or [38] and further emphasizes the generality of our system.

5. CONCLUSIONS

This paper proposed a novel speaker diarization system SphereDiar. The system includes two neural networks and one clustering algorithm: SphereSpeaker neural network for speaker embedding extraction, HBS neural network for segmentation and Top Two Silhouettes for clustering. In our experiments, we focused on evaluating the system with 225 meetings and illustrated that the system could be simplified by excluding HBS. Using the best system configuration, we achieved average DER over the meetings as 3%. We compared our system with two state-of-the-art speaker diarization systems and showed that the results obtained with our system were better.

Nevertheless, the system still suffers from deficiencies. Firstly, the dimension of the speaker embeddings is relatively large which slows clustering. Secondly, Top2S does not yet have any proper theoretical foundation. Furthermore, this algorithm is also not very suitable for situations were only few frames for each speaker can be attained. Finally, we have not presented any methods for SAD. In future work, we would like to address each of these shortcomings.

6. ACKNOWLEDGEMENTS

We would like to thank Anja Virkkunen and Stig-Arne Grönroos for their helpful comments. This work was supported by the European Unions Horizon 2020 research and innovation programme via the project MeMAD (GA780069). Computational resources were provided by the Aalto Science-IT project.
7. REFERENCES


