Unsupervised Fine-tuning of Speaker Diarisation Pipelines using Silhouette Coefficients

Lucas van Wyk$^{1[0000-0001-8254-4850]}$, Marelie H Davel$^{1[0000-0003-3103-5858]}$, and Charl van Heerden$^{2[0000-0002-3212-2147]}$

^{1} Faculty of Engineering, North-West University, South Africa
and CAIR, South Africa
lucvanwyk@gmail.com
^{2} Saigen, South Africa
http://engineering.nwu.ac.za/must

Abstract. We investigate the use of silhouette coefficients in cluster analysis for speaker diarisation, with the dual purpose of unsupervised fine-tuning during domain adaptation and determining the number of speakers in an audio file. Our main contribution is to demonstrate the use of silhouette coefficients to perform per-file domain adaptation, which we show to deliver an improvement over per-corpus domain adaptation. Secondly, we show that this method of silhouette-based cluster analysis can be used to accurately determine more than one hyperparameter at the same time. Finally, we propose a novel method for calculating the silhouette coefficient of clusters using a PLDA score matrix as input.

Keywords: Speaker Diarisation · Unsupervised Fine-Tuning · Domain Adaptation

1 Introduction

The goal of speaker diarisation is to identify different speakers in an input audio file, and to segment the input into speaker-homogeneous segments where each segment-level utterance is spoken by only one of the identified speakers. By attributing speaker identities to audio segments, speaker diarisation is used to automatically annotate audio [1] and to provide speaker-specific metadata to downstream automatic speech recognition (ASR) systems for speaker-specific adaptations, which have been shown to improve recognition performance [2].

Speaker diarisation pipelines are often modular systems and can be divided into two main components: segmentation and clustering. Segmentation is a two-part process that divides an audio recording into smaller speech segments that likely contain a single speaker and extracts a speaker embedding, or representation, for each segment. The clustering step accepts the embedded segments as input and labels each segment according to speaker identity.

Currently, the standard approach for extracting speaker embeddings utilises time-delay neural networks (TDNNs) to extract speaker embeddings, known as x-vectors [3]. Extracting x-vectors involves training a TDNN to map speech features, such as Mel-frequency cepstral coefficients (MFCCs) or filter banks (Fbanks) to a fixed dimensional subspace. Upon extracting speaker embeddings, most diarisation approaches train a probabilistic linear discriminant analysis (PLDA) model to project speaker embeddings to a latent space where it is assumed that each speaker embedding is generated from a Gaussian centered around the speaker mean. PLDA models are used to determine the probability that two embeddings originate from the same or different speakers.

A common challenge for speaker diarisation and speaker recognition systems is domain adaptation, as a large amount of in-domain data is needed to sufficiently train an x-vector model. Furthermore, the PLDA backend also requires a large amount of data to sufficiently model speaker representations in a way that speaker representations are separable [4]. Finally, the number of speakers, or clusters, in an input audio file is generally unknown and has to be determined.

In this paper, we implement a speaker diarisation pipeline on real-world call centre data. We focus on how the speaker embedding and PLDA models, which were pre-trained on out-of-domain data, can be adapted for our in-domain use case. Lastly, we show how to perform cluster analysis using silhouette coefficients for the dual purpose of determining the number of speakers in a recording and fine-tuning the domain adaptation process in an unsupervised, per-file manner.

The rest of the paper is structured as follows: Section 2 provides background on the techniques used in this study before discussing closely related work (Section 3). The study is conducted in the context of a fairly standard diarisation pipeline, as described in Section 4. Section 5 describes our proposed approach to unsupervised fine-tuning and Section 6 describes our dataset, experiments and results. Final conclusions are briefly summarised in Section 7.

2 Background

In this article we mainly focus on the use of silhouette coefficients and PLDA domain adaptation in the context of speaker diarisation.

2.1 Silhouette Coefficients

Silhouette coefficients measure the quality of data clusters by comparing the distance between datapoints in the same cluster (intra-cluster distance) to the distance between datapoints in different clusters (inter-cluster distance) [5]. To calculate the silhouette coefficients of data clusters, we first define the average intra-cluster and inter-cluster distances of a datapoint i as $a(i)$ and $b(i)$ respectively, where:

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i} d(i, j) \tag{1}$$

and

$$b(i) = \min_{k \neq i} [\frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)] \tag{2}$$

Here C_i is the cluster that datapoint i belongs to and $|C_i|$ is the size of cluster C_i; $d(i,j)$ is the distance between datapoints i and j. C_k represents some other cluster than C_i with i and k in $\{1...N\}$. N is the total number of datapoints in the dataset. Thus, $a(i)$ is the average distance between datapoint i and all other datapoints in C_i and $b(i)$ is the average distance between datapoint i and all other datapoints in the closest cluster. The silhouette coefficient of datapoint i is now defined as:

$$s(i) = \frac{b(i)-a(i)}{max(a(i),b(i))}, \text{ if } |C_i| > 1 \tag{3}$$

with $-1 \leq s(i) \leq 1$. Since more than one datapoint is needed per cluster to calculate a silhouette coefficient, $s(i)$ is undefined in cases where $|C_i| = 1$. The silhouette coefficient over an entire dataset is defined as:

$$SC = \bar{s}(n) \tag{4}$$

Here $\bar{s}(n)$ represents the mean silhouette coefficient over the entire dataset which is then used to evaluate cluster quality; an SC (silhouette coefficient) value of 1 represents perfect clusters: the clusters have an arbitrary inter-cluster distance and an intra-cluster distance of 0. Conversely, an SC value of -1 represents no clustering i.e. clusters overlap completely and no distinction is made between classes. From this point onward we define the silhouette coefficient as SC, calculated as shown in equation 4.

2.2 Probabilistic Linear Discriminant Analysis

The goal of the PLDA backend, as used in speaker diarisation, is to statistically model or 'explain' the distributions of speaker embeddings as a function of speaker identity. Although many variations of PLDA models exist, we use Gaussian PLDA models, which models speaker distribution as a mixture of Gaussian distributions [6]. A Gaussian PLDA model can be conceptualised as a Gaussian Mixture Model (GMM) with the equation:

$$P(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\mathbf{y}, \theta_W) \tag{5}$$

Here $P(\mathbf{x}|\mathbf{y})$ is the probability of observing the speaker embedding $\mathbf{x}$ given the latent variable $\mathbf{y}$ with $\mathbf{y}$ representing speaker identity and also being the centre of the mixture component. θ_W represents the 'within-class' covariance matrix. Additionally we assume that the latent variable $\mathbf{y}$ is modelled by:

$$P(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{m}, \theta_B) \tag{6}$$

where $\mathbf{m}$ is the mean over the entire dataset and θ_B is the 'between class' covariance matrix.

The PLDA model then applies a linear transformation A to the speaker embeddings such that $\theta_W = AA^T$ and $\theta_B = A\Psi A^T$ [6], resulting in the following model:

$$\begin{aligned} &\mathbf{x} = \mathbf{m} + A\mathbf{u} \\ &\mathbf{y} = \mathbf{m} + A\mathbf{v} \\ &\mathbf{u} \sim \mathcal{N}(\cdot|\mathbf{v}, \mathbf{I}) \\ &\mathbf{v} \sim \mathcal{N}(\cdot|0, \Psi) \end{aligned} \tag{7}$$

Once estimated, the PLDA model can be used to generate a similarity score between two speaker embeddings; this similarity score is known as a PLDA score. The PLDA score, denoted as R is nothing more than the ratio of the likelihoods of two speaker embeddings belonging to the same class as opposed to different classes. We can then use the PLDA scores to construct a PLDA score matrix which is a square matrix with each column representing the PLDA scores between a specific speaker embedding and every other speaker embedding.

Lastly, a Gaussian PLDA model makes two assumptions: the first being that speaker embeddings follow a Gaussian distribution and the second assumption being that the dimensions of speaker embeddings are statistically independent or, in other words, θ_W and θ_B are diagonal and speaker embedding dimensions are uncorrelated both within and between classes. Dehak et al. [7] shows that speaker recognition systems can greatly benefit from transforming speaker embeddings to a subspace where the independence assumption holds.

In order to transform the speaker embeddings into a space that satisfies the assumptions imposed by the PLDA model we apply a centering and whitening transform to speaker embeddings, as proposed in [8]. In practice, the centering and whitening transformation is obtained by performing Linear Discriminant Analysis (LDA) on speaker embeddings as in [6] and [7]. LDA defines a new spatial axis which maximises the variance between speakers and minimises the variance within speakers and, as a result, simultaneously diagonalises the θ_W and θ_B matrices. In addition to applying the centering and whitening transform obtained from LDA, the speaker embeddings are length normalised using L2-norm. Once LDA and length normalisation are applied to the speaker embeddings the PLDA backend can be trained.

2.3 PLDA Adaptation

In [9] it is shown that speaker recognition models can be sufficiently adapted for new domains by adapting the parameters of the PLDA model, specifically θ_W and θ_B. To adapt the parameters of a PLDA model we require labelled data, which provides information to learn θ_W which characterises channel distortions and θ_B which characterises speaker information. In this article we focus on PLDA interpolation, which is a domain adaptation technique wherein two PLDA models are needed: The first PLDA model is trained on large amounts of out-of-domain data and the second PLDA model is trained on a small labelled in-domain subset. The 'within-class' and 'between-class' covariance matrices, θ_W and θ_B of both the in-domain and out-of-domain PLDA models are then interpolated to form a new adapted PLDA model. The PLDA interpolation is defined as:

$$\begin{aligned} &\theta_{W-adapt} = \alpha\theta_{W-in} + (1-\alpha)\theta_{W-out} \text{ and} \\ &\theta_{B-adapt} = \alpha\theta_{B-in} + (1-\alpha)\theta_{B-out} \quad \text{with} \\ &\alpha \in [0,1] \end{aligned} \tag{8}$$

Where $\theta_{W-adapt}$ is the adapted 'within-class' covariance matrix and θ_{W-in} and θ_{W-out} are the in-domain and out-of-domain covariance matrices respectively. Likewise $\theta_{B-adapt}$ is the adapted 'between-class' covariance matrix and

θ_{B-in} and θ_{B-out} are the in-domain and out-of-domain covariance matrices respectively. The size of the interpolation parameter, α is proportional to the influence of the in-domain PLDA model during adaptation.

Since different domains have different speaker and channel effects it is common practice to retrain the LDA and PLDA model on the new domain, however, sufficiently training a PLDA model requires large amounts of data [9], which is why we perform PLDA interpolation as explained in Section 8. The procedure for adapting an out-of-domain system for in-domain data is as follows:

1. Train a LDA transformation using labelled in-domain data.
2. Train a PLDA model on out-of-domain data using the in-domain LDA.
3. Train a PLDA model on labelled in-domain data using the in-domain LDA transformation.
4. Interpolate the out-of-domain and in-domain PLDA models as in Equation 8 to obtain an adapted PLDA model.

The de facto method for estimating the optimal α parameter for PLDA interpolation, in the case of speaker recognition, is to evaluate different values for α on a small held-out set [9]. While this method is sufficient for speaker recognition it is not sufficient for our use case as we only have a small amount of labelled in-domain data and, unlike in speaker recognition, speaker utterances are not clean, fixed-length utterances.

3 Related Work

Our goal is to adapt a pre-trained speaker diarisation system to a new domain. Ideally, one would want to re-train an x-vector and PLDA model on in-domain data which would negate the need to use out-of-domain, pre-trained models. However, a large amount of in-domain data is required to sufficiently train x-vector and PLDA models [3], [4], which is why we focus on adapting pre-trained PLDA models for new domains. Additionally, sufficiently adapting a PLDA model for in-domain data enables us to use a pre-trained x-vector model as-is without any additional training. We aim to use the SC, calculated from the output of the diarisation system, together with PLDA interpolation for the joint purposes of adapting a pre-trained PLDA model for in-domain data and determining the number of speakers in a recording.

SC, along with spherical K-Means [5] have been shown to be useful in determining the correct number of clusters or, in the case of speaker diarisation, speakers in an input audio file. In Kaseva et al. [10] the SC is used, in conjunction with spherical K-Means to empirically determine the number of speakers in an audio file by iteratively performing K-Means with different numbers of clusters and comparing the SC of each iteration to determine the correct number of speakers.

In terms of unsupervised adaptation it was shown in [9] and [11] that PLDA models can be iteratively adapted on unlabelled in-domain data by performing diarisation on in-domain data using an out-of-domain system and iteratively interpolating the out-of-domain PLDA using a PLDA trained on in-domain data which uses the speaker clusters as labels.

In this work, we propose using SC for finding the optimal interpolation parameter α and number of speakers in an unsupervised way and on a per-file basis. To our knowledge, this work is the first implementation of SC in PLDA interpolation, as well as the first attempt at applying PLDA interpolation on a per-file rather than per-corpus basis.

4 System Description

In this section we describe the different parts of our speaker diarisation system, namely speech activity detection, speaker embeddings, clustering and scoring. The layout and order of components in the system are illustrated in Figure 1.

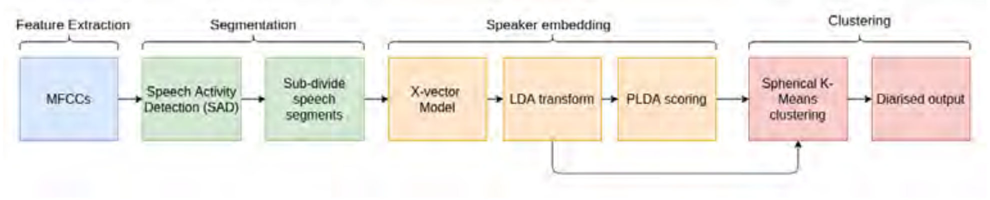

Fig. 1. Block diagram of the speaker diarisation pipeline, indicating the order of components.

4.1 Speech Activity Detection

For speech activity detection we develop our model using a popular process described in the ASPIRE kaldi recipe[3]. The model consists of a TDNN that accepts 40-dimensional MFCCs extracted every 30ms using a 25ms window; these are fed to a TDNN consisting of 5 TDNN layers followed by 2 statistical pooling layers. In an analysis not shown here, we had also tested other publicly available SAD models such as WebrtcVAD [12] and PyAnnote-SAD [13] but found the TDNN model to deliver the best results.

4.2 Speaker Embedding Architecture

To obtain speaker embeddings, we use a pre-trained speaker embedding model[4]. The speaker embedding model is a TDNN which is based on the original TDNN architecture as shown by Snyder et al. [3] but has 13 layers instead of 9. The TDNN and its PLDA backend was trained for speaker recognition on 16kHz speech from the Voxceleb1 and Voxceleb2 training sets with an additional 51,200

[3] `https://github.com/kaldi-asr/kaldi/tree/master/egs/aspire/s5`
[4] `https://github.com/Jamiroquai88/VBDiarization`

segments obtained through data augmentation using noise sampled from the MUSAN corpus and simulated Room Impulse Responses (RIR) [14].

The TDNN network accepts 23-dimensional mean-normalised MFCCs extracted every 25ms with a 10ms shift and was trained for 9 epochs using categorical cross entropy loss. The model architecture is shown in Table 1. The first 10 layers operate on speech frames with a small temporal context centred at frame t, and the statistical pooling layer aggregates the frame-level outputs from layer 10, and computes the mean and standard deviation. The statistical pooling layer aggregates the information across the time dimension so that subsequent layers operate on the entire segment. The mean and standard deviation computed by the statistical pooling layer are concatenated together and propagated through segment-level layers to the output softmax layers. All non-linearities are rectified linear units (ReLUs). During inference, we extract x-vectors using a 2 second sliding window with a 250ms step size as we found this segmentation to deliver the best results (in an analysis not shown here).

Table 1. The pre-trained TDNN architecture as described in [14]. K represents the input feature dimensionality, T is the number of segments and N is the number of speakers.

Layer	**Layer context**	**(Input) x output**
frame 1	[t-2,t+2]	(2xK) x 512
frame 2	[t]	512 x 512
frame 3	[t-2,t,t+2]	(3x512) x (512)
frame 4	[t]	512 x 512
frame 5	[t-3,t,t+3]	(3x512) x 512
frame 6	[t]	512 x 512
frame 7	[t-4,t,t+4]	(3x512) x 512
frame 8	[t]	512 x 512
frame 9	[t]	512 x 512
frame 10	[t]	512 x 1500
stats pooling (mean + stddev)	[0,T]	(2x1500) x 512
segment1	[0]	512 x 512
softmax	[0]	512 X N

4.3 Spherical K-Means Clustering

We use a combination of vanilla and spherical K-Means to cluster speaker embeddings according to speaker identity. We found this method of K-Means clustering to deliver better results than other popular clustering methods such as agglomerative hierarchical clustering (AHC) [15] and variational Bayes hidden Markov model clustering (VBHMM) [16]. Our clustering approach accepts the extracted x-vectors and PLDA score matrix as input and the exact implementation is as follows.

1. Apply the LDA transformation and L2-norm length normalisation to all speaker embeddings.

2. Perform spherical K-Means. The centroids are intialised randomly.
3. Using the PLDA score matrix as input, perform vanilla K-Means with the PLDA scores being the distance between two x-vectors. We use the centroids from the spherical K-Means step as initialisation.

After clustering, each segment belonging to the same cluster is merged and if two adjacent, overlapping segments belong to the same cluster, the speaker change point is taken as the midpoint between the two segments as shown by Landini et al. [16].

4.4 Diarisation Error Rate

The accuracy of a speaker diarisation system is measured using the diarisation error rate (DER) [17] where DER is defined as:

$$DER = \frac{False\,Alarm + Missed\,Detection + Speaker\,Confusion}{Total\,Duration\,of\,Time} \tag{9}$$

False alarm refers to the total time labelled as speech which is actually non-speech, missed detection refers to the total time labelled as non-speech that contains speech, and speaker confusion refers to the total time labelled as belonging to the wrong speaker. When scoring, a Hungarian Algorithm [17] is employed to establish a one-to-one mapping between the hypothesis and reference outputs. The DER metric also allows for a collar (usually 0.25 seconds) set around every boundary of the reference segments where the output will not be scored. A collar is used to account for inconsistent annotation errors in the reference transcript [17]. In this work, we use a 250ms collar for all experiments.

5 Unsupervised Fine-tuning

As mentioned in Section 2.2, the optimal α parameter for PLDA interpolation for use in speaker recognition is typically obtained using a small held-out set. However, in diarisation, the optimal α parameter for an entire corpus does not necessarily indicate the optimal α for each audio file, especially if audio files within a corpus have substantial differences in terms of recording and noise conditions. Furthermore, with unseen or unlabelled data the number of speakers in a recording is not always known and has to be determined (for use in the K-Means clustering step). In this paper we perform unsupervised fine-tuning by using the SC to not only find the correct number of speakers but also the optimal α parameter for each recording.

The procedure for unsupervised fine-tuning is as follows:

1. Perform diarisation on each recording over a range of different values for both the α parameter and number of speakers (or clusters).
2. Calculate SC for each combination of α and number of speakers.
3. For each recording, identify the α and number of speakers with the highest SC.

In our experiments we compare two methods of computing the SC: The first method follows the standard modus operandi and uses the cosine distance between speaker embeddings to compute the average inter- and intra-cluster distances ('standard SC'). The second method uses the cosine distance between the columns of the PLDA score matrix to compute the average inter- and intra-cluster distances ('score matrix SC'). The intuition behind our second method is that, if the cosine distance between the n^{th} and m^{th} column of the PLDA score matrix is small, the n^{th} and m^{th} speaker embeddings have similar PLDA scores with respect to other embeddings and are therefore similar to each other and likely belong to the same speaker.

6 Analysis

6.1 Dataset

For all experiments reported on in Section 6.3 we use a proprietary corpus consisting of telephone calls from 9 different South African call centres containing calls in English, Sotho and isiZulu. The corpus consists of 118 calls spanning 20.13 hours and differs from the original training data in many aspects, such as background noise, sampling rate and encoding. The audio is originally recorded in stereo format, one channel for the call centre agent(s) and the other for the customer(s) and saved in a mono WAV49 format which is an 8kHz signal passed through a GSM codec and under WAV encapsulation. However, for diarisation we upsample the audio to 16kHz prior to feature extraction using sox[5] as the pre-trained models were trained on 16kHz audio.

Table 2 shows the composition of calls in the corpus, which we refer to from here onwards as the SATS (South African Telephone Speech) corpus.

Table 2. The number of calls and duration of total calls for each call centre in the SATS corpus.

	Calls	**Hours**
Call Centre 1	19	1.95
Call Centre 2	9	1.96
Call Centre 3	6	1.21
Call Centre 4	5	1.43
Call Centre 5	17	4.66
Call Centre 6	4	2.41
Call Centre 7	32	3.82
Call Centre 8	17	1.25
Call Centre 9	9	1.44
	118	20.13

[5] `http://sox.sourceforge.net/`

6.2 Experimental setup

Prior to performing any domain adaptation we split the SATS corpus into an 80-20 development and test set with both sets containing calls from each call centre.

For all the approaches we compare, we follow the same initial process: We perform LDA on the development set of the SATS corpus. Using this in-domain LDA transform we train two PLDA models: an out-of-domain PLDA trained on the Voxceleb1 and Voxceleb2 datasets with data augmentation (as explained in Section 4.2) and an in-domain PLDA model trained on the development set. After training we interpolate the covariance matrices of the out-of-domain and in-domain PLDA models as shown by Equation 8. We then determine both the number of speakers and interpolation parameter in three different ways: using a supervised approach as benchmark, and comparing this with two variations of the unsupervised technique introduced in Section 5.

First, we obtain a baseline in a *supervised* fashion, that is, we use the DER over all calls to guide the process of selecting α. For the number of speakers, we use the known ('oracle') value. Specifically, to estimate the optimal α parameter, we perform diarisation on the development set for $\alpha \in [0.5, 1.0]$, providing the oracle number of speakers for each call. We then choose the α with the lowest corresponding average DER over all calls ('per-corpus fine-tuning').

For unsupervised fine-tuning the value of α and number of speakers are jointly estimated by performing diarisation on the development set for α in the range $\alpha \in [0.5, 1.0]$ and speakers in the range $N_{speakers} \in [2, 6]$. We then choose the combination of α and number of speakers resulting in the highest SC for each call ('per-file fine-tuning'). That is, in addition to detecting the number of speakers we obtain an optimal interpolated PLDA model for each call as opposed to one interpolated PLDA model for the entire corpus. The two different fine-tuning approaches we compare relate only to the distance measure used when calculating the SC: either using the cosine distance between speaker embeddings ('standard SC') or the cosine distance between the columns of the PLDA score matrix ('score matrix SC').

6.3 Results

Table 3 shows the average DER taken over all calls as well as the relative improvement of our fine-tuning approaches compared to diarisation without domain adaptation (that is, diarisation performed using the pre-trained system as-is). Notice that using SC with the PLDA score matrix as input performs very similarly to the use of SC with only the speaker embeddings as input. Furthermore, although the performance difference between unsupervised fine-tuning using speaker embeddings and the PLDA score matrix is marginal on the development set, we notice that unsupervised fine-tuning with speaker embeddings performs noticeably better on the test set. This is somewhat expected as domain mismatch between the development and test sets would cause the performance of the PLDA model and LDA transformation to degrade, allowing for less discriminatory information in the PLDA score matrix.

Table 3. The average DER over all calls for the development and test sets using supervised per-corpus and unsupervised per-file fine-tuning. Also shown is the relative average DER improvement ('Relative') of fine-tuning compared to diarisation without any domain adaptation (using the pre-trained system as-is). DER is calculated using a 250ms collar.

	Development Set		Test Set	
	DER	Relative	DER	Relative
Per-corpus fine-tuning, Oracle Speakers	12.03	8.59	12.67	25.07
Per-file fine-tuning, Score matrix SC	11.27	14.36	12.44	26.43
Per-file fine-tuning, Standard SC	**11.25**	**14.51**	**12.31**	**27.20**

Figure 2 compares the average DER obtained with and without fine-tuning on the development and test sets. The blue and orange dotted lines represent the best achievable average DER for the development and test sets respectively (i.e. the combination of α and number of speakers that achieved the lowest DER for each call in the set). Notice that the performance gain provided by the unsupervised per-file fine-tuning decreases on the test set for both approaches. This can be accredited to domain mismatch between the development and test sets which degrades the effectiveness of the LDA and PLDA. The effects of domain mismatch between calls may be alleviated by training the LDA and in-domain PLDA on a larger test set. The probability distributions of the DER for each method can be seen in Figures 5 and 6.

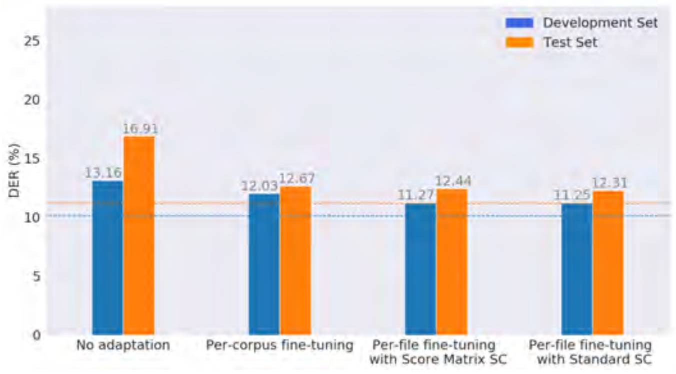

Fig. 2. Average DER measured on the development and test using no adaptation (i.e., using the pre-trained system 'as-is'), supervised per-corpus fine-tuning and unsupervised per-file fine-tuning using 'Standard SC' and 'Score Matrix SC'. The dotted lines shows the best achievable average DER over all runs. DER is calculated using a 250ms collar.

Figure 3 shows the number of calls with n speakers (blue) versus the number of calls with n speakers as predicted by unsupervised fine-tuning with and without the PLDA score matrix as input (orange and green, respectively) for the development set; Figure 4 shows the same results for the test set. Notice that the unsupervised per-file fine-tuning tends to underestimate the number of speakers in a recording. This result makes intuitive sense: K-Means clustering is generally sensitive to inbalanced datasets; if a call has multiple speakers but one speaker is active for the majority of the time K-Means may classify the majority speaker(s) better when forming a smaller amount of clusters. Lastly, the DER of a call may also be decreased by underestimating the number of speakers if it leads to better classification of majority speakers.

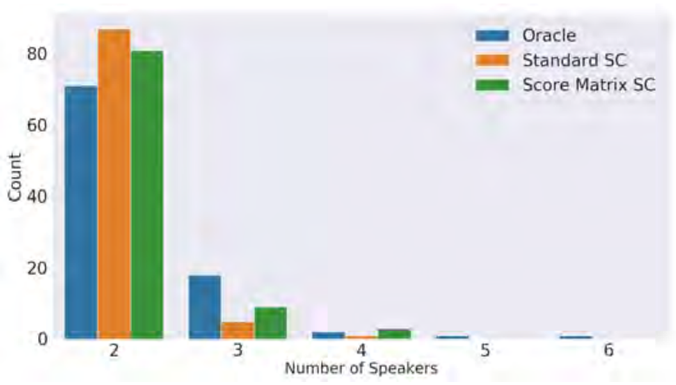

Fig. 3. True ('Oracle') versus predicted number of calls containing a specific number of speakers using unsupervised fine-tuning on the development set of the SATS corpus.

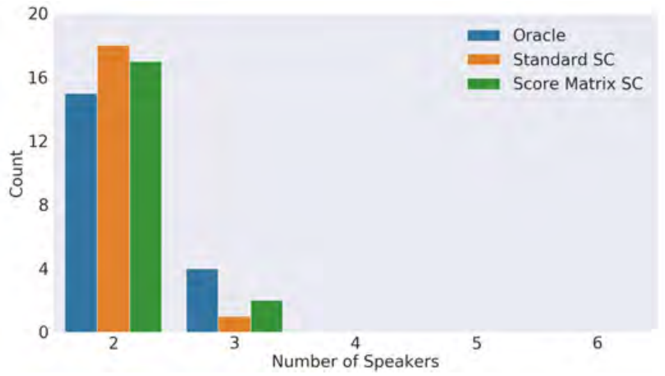

Fig. 4. True ('Oracle') versus predicted number of calls containing a specific number of speakers using unsupervised fine-tuning on the test set of the SATS corpus.

7 Conclusion

In this paper we demonstrate the effectiveness of using the SC of a diarisation output to fine-tune the interpolation parameter for PLDA models when performing domain adaptation, while simultaneously determining the number of speakers in a call. Furthermore, we show that by adapting and fine-tuning a speaker diarisation system on a per-file basis we can substantially increase the performance of a pre-trained speaker diarisation system.

Additionally, we proposed calculating the SC with the PLDA score matrix, instead of the speaker embeddings, as input and show that this method has comparable performance to the alternative. Although using speaker embeddings as input had ultimately shown better performance, the PLDA score matrix provides a method of evaluating the robustness of PLDA models in domain mismatch conditions.

Lastly, we show that unsupervised fine-tuning has a tendency to underestimate the number of speakers in a call. Although this underestimation can in some cases actually improve the DER, it may affect the performance of downstream components such as ASR systems which use speaker annotations for speaker-specific adaptations. For this reason one may rather seek to fine-tune diarisation systems based on the Jaccard Error Rate (JER) rather than the DER as the JER scores diarisation output on a per-speaker basis [18].

Acknowledgement

The authors acknowledge the Centre for High Performance Computing (CHPC), South Africa, for providing computational resources for this project.

References

[1] D. Haws, D. Dimitriadis, G. Saon, S. Thomas, and M. Picheny, "On the importance of event detection for ASR," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2016, pp. 5705–5709.

[2] P. Cerva, J. Silovsky, J. Zdansky, J. Nouza, and L. Seps, "Speaker-adaptive speech recognition using speaker diarization for improved transcription of large spoken archives," *Speech Communication*, vol. 55, no. 10, pp. 1033–1046, 2013.

[3] 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.

[4] D. Garcia-Romero, A. McCree, S. Shum, N. Brummer, and C. Vaquero, "Unsupervised domain adaptation for i-vector speaker recognition," in *Proceedings of Odyssey: The Speaker and Language Recognition Workshop*, vol. 8, 2014.

[5] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

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

[7] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification," in *Tenth Annual conference of the international speech communication association*, 2009.

[8] D. Garcia-Romero, "Robust speaker recognition based on latent variable models," Ph.D. dissertation, 2012.

[9] D. Garcia-Romero and A. McCree, "Supervised domain adaptation for i-vector based speaker recognition," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2014, pp. 4047–4051.

[10] T. Kaseva, A. Rouhe, and M. Kurimo, "Spherediar: An effective speaker diarization system for meeting data," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, IEEE, 2019, pp. 373–380.

[11] G. Le Lan, D. Charlet, A. Larcher, and S. Meignier, "Iterative plda adaptation for speaker diarization," in *Interspeech 2016*, vol. 2016, 2016, pp. 2175–2179.

[12] *Webrtcvad*, `https://webrtc.org/`, Accessed: 2021-09-23.

[13] M. Lavechin, M.-P. Gill, R. Bousbib, H. Bredin, and L. P. Garcia-Perera, "End-to-end Domain-Adversarial Voice Activity Detection," in *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain, May 2020.

[14] J. Profant, *Robust speaker verification with deep neural networks*, 2019. [Online]. Available: `https://www.vutbr.cz/en/students/final-thesis/detail/122072`.

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

[16] F. Landini, J. Profant, M. Diez, and L. Burget, "Bayesian hmm clustering of x-vector sequences (vbx) in speaker diarization: Theory, implementation and analysis on standard tasks," *Computer Speech & Language*, vol. 71, p. 101 254, 2022.

[17] J. G. Fiscus, J. Ajot, M. Michel, and J. S. Garofolo, "The rich transcription 2006 spring meeting recognition evaluation," in *International Workshop on Machine Learning for Multimodal Interaction*, Springer, 2006, pp. 309–322.

[18] N. Ryant, K. Church, C. Cieri, J. Du, S. Ganapathy, and M. Liberman, "Third dihard challenge evaluation plan," *arXiv preprint arXiv:2006.05815*, 2020.

A Appendix

Figures 5 and 6 shows the probability density functions for the diarisaiton error rates obtain on the SATS development and test sets using different methods of adaptation compared to the best error rate obtained over all runs (blue). Notice that, for both the train and test sets the highest density is between the 0 and 15% error margins which coincides with the average DER reported in Table 3.

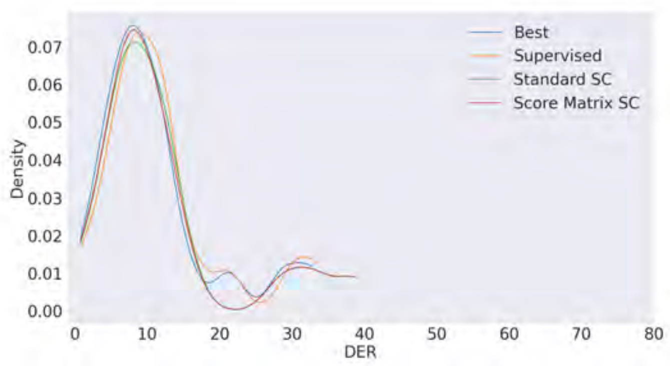

Fig. 5. Probability density function of the diarisation error rates obtained on the development set of the SATS corpus using different adaptation methods.

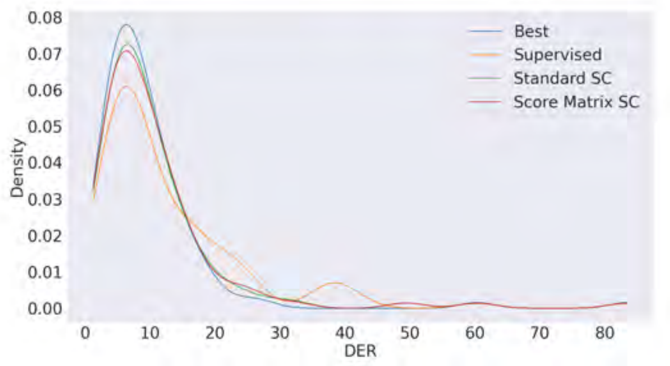

Fig. 6. Probability density function of the diarisation error rates obtained on the test set of the SATS corpus using different adaptation methods.