Speaker Diarization with Subtitles

Patricico Foncea foncea@mit.edu
Andreea Georgescu andreeag@mit.edu
Andrey Sushko asushko@g.harvard.edu

Abstract

Drawing techniques from unsupervised and supervised learning, we implement and compare different methods for speaker diarization when there is partial knowledge over the segmentation of speakers and a small portion of labeled data. Our approach can be roughly divided in three relatively independent parts: sound processing and segmentation, feature selection and dimensionality reduction, and clustering. Finding optimal combinations of algorithms for each of these components we achieve a diarization error of 13% over the *Monty Python's Flying Circus* dataset. However, our main contribution is a hands-on study on the peculiarities of this problem and the efficacy of some machine learning tools frequently used in this context.

1 Introduction

Speaker diarization is the process of labeling a speech signal with labels corresponding to the identity of speakers (Moattar and Homayounpour, 2012); in other words, answering the "who spoke when" question. The problem of identifying speakers appears naturally in the context of automatic speech transcription and in many other related contexts, with many applications in broadcast news, movie analysis, meetings transcriptions, conversational telephone scripts, etc. In this work we study a particular setting of this problem that has not been considered by previous work. In particular, we considered *partially segmented* data, which give us more information, but also sets more challenges in terms of the methods we can use.

1.1 Monty Python's Flying Circus

Our dataset consists of both the subtitles and the audio from almost all of the *Monty Python's Flying Circus* television show[1]. The audio sample is roughly 24 hours long and has 5 primary speakers, though some actors employ several distinct voices across different sketches. The subtitle files are formatted in *.srt* extension; using this information, and after reformatting the data, we produce a dataset where each row contains spectral data from a unique complete or partial sentence spoken by one speaker, delimited using the subtitle time window in which it was spoken (up to a resolution of 0.1s). The rest of the details of the data processing will be explained in the following section.

1.2 Literature Review

Speaker diarization is a fairly well studied problem, but developments are still in progress. The first approaches to tackle this problem were taken using unsupervised methods, but in recent years other methods have been considered. Previous works that have taken an approach closer to our have exploited a variety of clustering algorithms (Kotti et al., 2008) including k-means (Dimitriadis and Fousek, 2017; Wang et al., 2018), hierarchical clustering (Garcia-Romero et al., 2017; Sell and Garcia-Romero, 2014), spectral clustering (Wang et al., 2018; Ning et al., 2006), and Gaussian mixture models (Zajıc et al., 2017; Shum et al., 2013). More recent approaches include supervised deep neural networks (Zhang et al., 2018; Garcia-Romero et al., 2017), LTSM (Wang et al., 2018), convolutional neural networks (Zajıc et al., 2017) and d-vectors (Wan et al., 2018), among others. For a more complete review of the topic, the reader can refer to (Moattar and Homayounpour, 2012), (Anguera et al., 2012), and (Tranter and Reynolds, 2006).

2 Data Processing

Our data consists of three parts: subtitle files in the *.srt* format, audio from the episodes converted to

[1] *Monty Python's Flying Circus* is a British sketch comedy series created by the comedy group Monty Python and broadcast by the BBC from 1969 to 1974.

.wav format, and a selection of sketches for which correct speakers were manually assigned.

2.1 Subtitles

Subtitle files were obtained for all 45 episodes. An example of one entry is provided below.

> 452
> 00:23:08,922 $--$ $>$ 00:23:09,890
> I don't know anything.

The consistent format allows for straightforward parsing to extract, for each line, a line number, start and stop time, and the words spoken in that interval. Only the line numbers and start/end times were used for diarization, with the text serving purely for verification purposes.

2.2 Audio

Audio data for 41 out of 45 episodes was obtained from *YouTube*, converted from stereo *.mp3* to mono *.wav* format, and processed using the *scipy.signal* library. In order to classify speakers, we are not concerned with the pure audio signal, but rather with extracting certain conserved characteristics of the speech such as the pitch or cadence. We, therefore, begin by Fourier transforming the waveforms into spectrograms, in order to analyze the distribution of frequencies within the audio. Since the subtitle files already provide us with a significant part of the diarization algorithm by delimiting the times at which speakers may change, we seek to classify speakers using the time-averaged characteristics of each "line" of speech. This dramatically reduces the size and complexity of the feature vector by allowing us to collapse the 3-dimensional *(power, frequency, time)* data into 2-d *(mean power, frequency)*. Doing this effectively erases the content of the speech of each line, along with any information about cadence or rate of speech, however, it makes it significantly easier to compare lines of differing length and content, allowing for the analysis methods described in following sections. Losing the speech content may also be advantageous in scenarios where one wishes to collect spectral data for analysis without collecting potentially confidential content. The time and frequency resolution of the spectral data in determined by the Fourier transform window, with a short window providing high temporal resolution and a long window higher frequency resolution. Since we are time-averaging

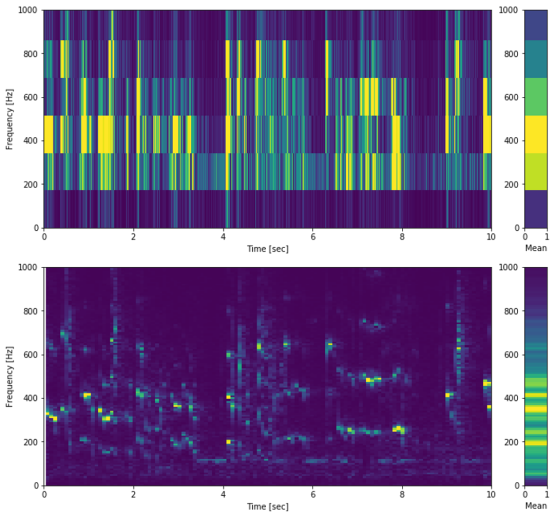

Figure 1: Amplitude spectrograms taken with a 6ms window (top) and 110ms window (bottom). Time-averages of both are shown on the right

the data, we make the trade towards higher frequency resolution by increasing the window up to the quantization level of the subtitle timing data (0.1 seconds). Figure 1, shows spectrograms for 10 seconds of speech generated using a default 6ms window and the 110ms window used for the subsequent analysis. Note that the latter reveals significantly more of the frequency-structure of the data within the $100Hz - 1kHz$ range in which most of the signal is found. Finally, to enable better direct comparison, the time-averaged spectrograms for each line have their amplitudes normalized so that the volume of the speech does not factor into the clustering distance metrics.

2.3 Subtitle-Audio Alignment

In order to verify the alignment of audio and subtitle data, a script was created to display the content of a given subtitle line and play the audio snippet corresponding to the start/end times provided in the subtitle data. Good alignment was achieved after globally re-scaling the subtitle timings by a factor of 1.176 owing to an apparent slowdown of the obtained audio data. A significant caveat is that even when timings are correctly aligned, there are instances in which the timing of a subtitle line does not match perfectly with the time in which the line is spoken, especially in cases where length of line varies significantly between speakers and adequate time must be given for the audience to read short words/phrases. In these cases, we expect partial mixing of different spectra within the

reported spectrum of that line, in effect blurring the boundaries between different clusters. These instances cannot be identified without being able to already perform speaker diarization, but occur sufficiently infrequently within the data that was manually sampled for us to just accept as a source of occasional error.

2.4 Verification

To verify clustering algorithms and perform supervised learning, a range of sketches had speakers manually labelled. In total, several hundred lines were analyzed, including a section of *S01E01*, the *Dead Parrot* sketch in *S01E08*, the *Fish License* sketch in *S02E10*, and the *Brontosaurus* sketch in *S03E05*.

Finally, to analyze the performance of our clustering algorithms we plot, for each true speaker, the frequency with which their lines are assigned a given label. Adjacently, we plot the number of lines spoken by the true speaker, to gauge the overall classification rates and visually highlight the cases in which the model is able or unable to distinguish between certain speakers. Since the ordering of labels is arbitrary, a script was implemented to rearrange them on some outputs for clarity. In those cases, labels are arranged such that perfect speaker identification would produce the identity matrix.

3 Unsupervised Clustering

Our initial approach was to use a simple k-means clustering algorithm. Evaluating the assignments on the labeled data samples of two speakers, we got very poor accuracy of around 55%, which, given the uneven split of lines between the two speakers, was not much better than a random guess.

When we consider the structure of our data, it immediately becomes clear why k-means is not well suited for our task. Our data points are amplitudes per frequency bin, averaged across complex sounds. Let's imagine we have three such data points: A comes from 30 seconds of a tenor's G_4, B from 30 seconds of the same tenor when he is off-key and singing slightly below a G_4, and point C from 30 seconds of a bass' A_1. Assume the three singers' amplitude is the same. Then, note that points B and C will have the same Euclidean distance from point A. However, point B is much more likely to come from the same singer as point A.

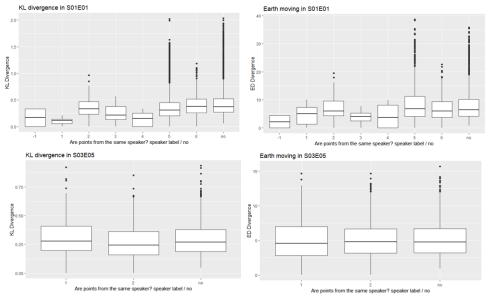

Figure 2: Pairwise distance per speaker, and across different speakers

To put it simply, our features or space coordinates have a meaningful order, which matters for our notion of *distance*. And this order makes the speaker-specific clusters non-spherical in Euclidean distance. Consequently, we decided to use Hierarchical trees with two custom distances, namely KL-divergence (averaged to become symmetric) and Earth Moving distance.

Let us first examine these custom distances. We will look at two samples of the labeled data, and compare the distribution of pairwise distance between points which are in the same cluster, and points which are not. If these distance metrics behave as we require, then the spread across pairs in the same clusters should be significantly lower compared to the spread of points in different clusters. We plot these distributions in Figure 2.

First, note that the two distance functions give similar results. Moreover, we observe that for some speakers the spread of the pairwise distances is consistent with our intuition, but for others it is not. Listening to the specific speakers, we can tell that the more distinctive voices correspond to the lower spread distributions in the plots above. This is reassuring in some sense, but suggests clustering will be difficult. We present our clustering results in the following section.

3.1 Hierarchical Tree Clustering

Given the plots in Figure 2, we choose the complete clustering method. In this method, we determine the distance between two clusters by taking the maximum distance between a pair across the clusters. We choose this method because in Figure 2, we notice that the difference between maximums seems more significant than that between averages, or minimums. Indeed, when we compare the complete method with other cluster-

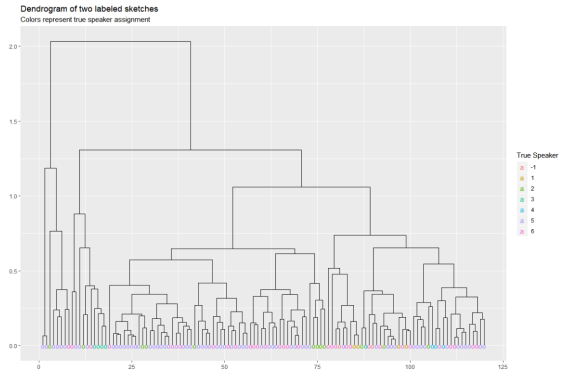

Figure 3: Dendrogram of the tree obtained using KL Divergence distance on two sketches in S01E01

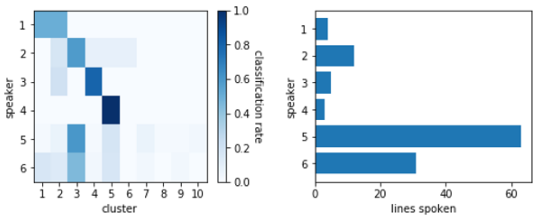

Figure 4: Performance of clustering method when tree is cut into 10 clusters

ing methods, we see similar or better results. We do not include the other methods in this report.

Let's first look at the dendrogram for the first labeled sample in Figure 2, *S01E01*. We include this in Figure 3.

We notice that speakers $2, 3$ and 4 are reasonably clustered together. However, speakers 5 and 6 appear very much randomly clustered. This is not surprising given Figure 2; however, it is clear that cutting the tree in 6 clusters, the initial number of speakers, will not give any meaningful speaker identification. Indeed, we notice that with only six clusters, we have a few very small clusters. We attribute this to the noisy nature of speakers 5 and 6 and cut the tree into 10 clusters instead.

We evaluate the performance by looking at a heatmap of true speakers being assigned to the new clusters, included in Figure 4.

The performance is in fact reasonable. Note that the extra clusters we allow in our model are virtually empty, which is consistent with our motivation to increase the number of clusters given some noise in the clustering tree. Again consistent with our observations so far, voices 3 and 4 are more distinguishable compared to $2, 5$ and 6.

Now, let us look at the dendrogram for the second sample in Figure 2, *S03E05*, a sketch labelled *Brontosaurus*. This plot is shown in Figure 5.

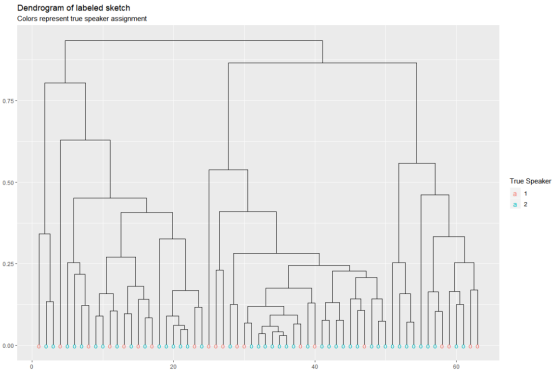

Figure 5: Dendrogram of the tree obtained using KL Divergence distance on *Brontosaurus* sketch

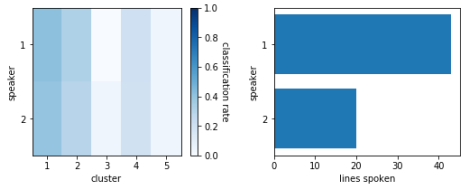

Figure 6: Performance of clustering method when tree is cut into 5 clusters

As we expected from Figure 2, the clustering in this case is virtually random. This conclusion also appears when we plot the heatmap performance in Figure 6.

To conclude this section, we expect that using a hierarchical tree clustering on the entire data set could perform relatively well in identifying the very distinguishable voices. As in the examples we presented, some exploration would be necessary to optimally choose the number of clusters. We believe that as the number of samples increases, the non-distinguishable voices will increase disproportionately to the distinguishable ones. Therefore, one would presumably need a lot more clusters than actual speakers in order to get good performance on the distinctive speakers. Nevertheless, as the majority of data-points come from indistinguishable speakers, we expect that the overall performance will be limited.

4 Supervised Approach

In Section 3 we presented a plot of the spread of pairwise distances of points which belong to the same speaker, and in contrast, which do not. What we noticed is that these spreads are often similar, which is not ideal for our clustering task.

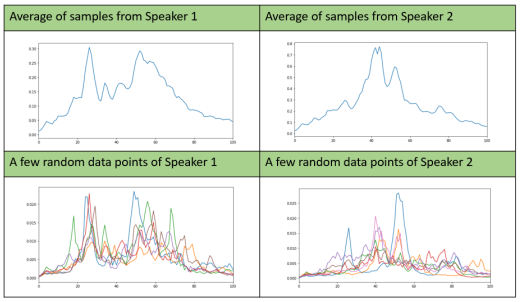

Figure 7: Average of samples compared to random data points from two speakers

Exploring the labeled data, we notice that the average of data points belonging to one speaker is generally distributed (across frequency bins) differently than the average of points of a different speaker. However, individual data points may look very similar. We include an example from the *S01E08 Dead Parrot* sketch in Figure 7.

This suggests that while the clusters behave differently as a whole, individual points may look very similar. With this observation in mind, we now re-formulate our problem into a supervised learning question. We can formulate the following problem: given a pair of data points, do they belong to the same speaker or not? On the labeled data we can learn to answer this question using logistic regression.

As we could expect, when we train our model on a random subset of the labeled data, and then validate the results on the remaining labeled data set, we get a high accuracy of 82%. However, if we use the data form one sketch to train the model, and test the accuracy on a different sketch, we get low accuracy of around 30%. This leads us to conclude that the labeled data is not representative, which is not surprising given the small sample.

5 PCA and Feature Selection

Given the nature of the data, it resulted natural to consider methods to reduce the number of columns or find the relevant components of the voice stamp. For this, we took two standard approaches: feature selection via correlation, and principal components analysis.

5.1 Correlated Frequencies

Computing the correlation matrix over the entire dataset for the 600 columns, we observe that many of them are highly correlated (over 90% of correlation), and that only a fifth of the column can be considered independent. More precisely, the first columns corresponding to low frequencies on the spectrum were found to be correlated with many of the high frequency columns. As a result of this, the new matrix consisted in most of the first tens of columns with an sparse selection of the higher ones. This is in opposition to our original hypothesis that contiguous frequencies would be the ones to be correlated and after doing feature selection we would end up with an uniformly sparse subset of columns.

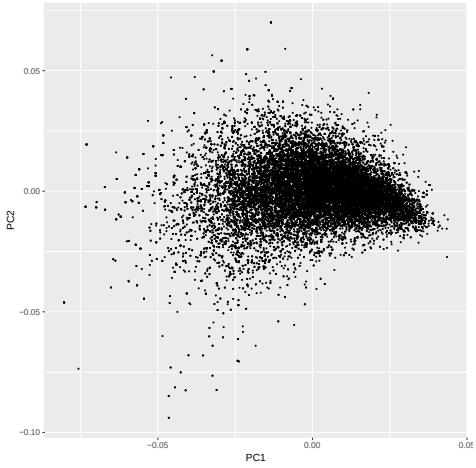

Figure 8: Coordinates of rows in first two components of PCA. No cluster structure can be obviously inferred.

5.2 Principal Component Analysis

After performing PCA, we can decrease the dimension of our problem to about 90% fewer columns and still explain 90% of the variation. We will see – with more detail in the next section –, however, that as with feature selection using correlations, this does not improve the quality of our model. To understand why this happens from a qualitatively point of view, consider Figures 8 and 9. From the first figure we can see that no cluster structure is obvious in the two most important components. The same holds for all pairs of components up to the sixth. In the second figure we see the result of PCA on one particular sketch with labeled data. There are two speakers represented by the two different colors. We observe a clear structure in the data suitable for classification, although this does not necessarily imply that there exist two well defined clusters.

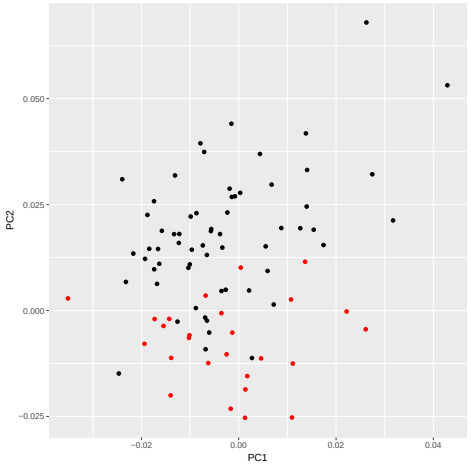

Figure 9: First two components of PCA on labeled data (*Dead Parrot*). Different colors correspond to two different speakers.

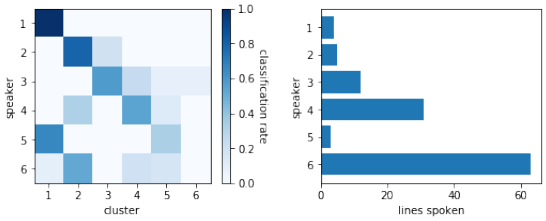

Figure 10: GMM on raw data.

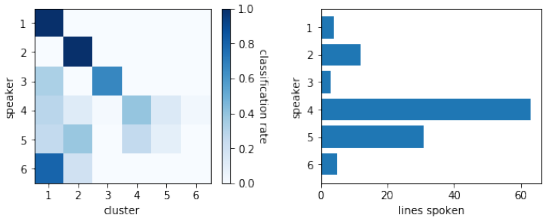

Figure 11: GMM on data after PCA.

6 Gaussian Mixture Model

As a last resort for clustering we fitted a Gaussian Mixture Model over the spectrogram data. This model assumes that the data is generated from one of k different Gaussian laws. This seems to be a plausible hypothesis for our data as the variations in the amplitude and frequency of our vocal registry can be thought as deviations from a central value that represents a neutral or ground state of our voice pattern.

We used this model under 3 different regimes. First, we applied it over the *raw* data (without any transformation or column reduction), over the data obtained after applying PCA, and over the data after doing feature selection. Surprisingly, at first sight at least, the best performance over the test labeled data is when the data is not processed. To better illustrate this, consider the analysis over a segment of *S01E01* where 6 different speakers are present. Figures 10 and 11 show the heatmap performance of the GM model over the raw data and the PCA transformed data. Note that the raw data is more successful identifying speakers, although both do a fairly decent job in finding the clusters.

The next step is to apply a mixture model without imposing the number of clusters. Since when applying the algorithm over the complete data set we will not necessarily know the number of speakers, it is important to test the performance in this case. When the number of speakers is small (two or three), the model manages to identify the correct number. However, as the number of speakers increases – especially if some of the speakers represent only a few lines –, the model does not necessarily capture that complexity. However, one notable feature is that when unconstrained, the model is better at identifying the *change in speaker*. This is particularly promising since this task is actually a very complex one. In Figure 12 we can see the heatmap for the unconstrained GM model. It merges pairs of speakers, distinguishing rather well among them, beyond the obvious fact that the number of cluster differs.

Finally, the most insightful observation was obtained when analyzing the results for the two person *Dead Parrot* sketch. First, we note that besides the high accuracy in identifying speaker (diarization error of 13%), we also obtained a fairly good model on *when* there is a change of speaker, which is a slightly more delicate question. In this case, we have a 40% error (changes that were not captured by the model) and a 40% rate of false positives (the model changes, but there is no real change). Secondly, as we mentioned earlier, projecting into the first two PCA components reveals some structure, but without adequate separation

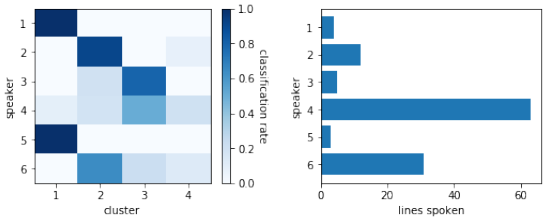

Figure 12: Unconstrained GMM on raw data.

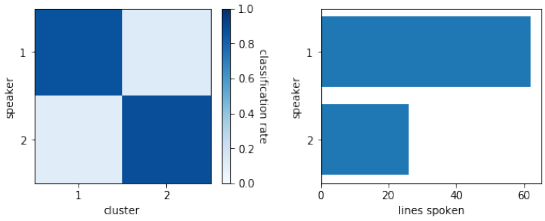

Figure 13: GMM on *Dead Parrot*. Raw data.

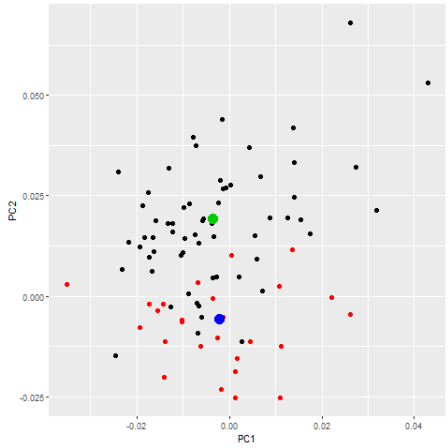

Figure 14: Centers of GMM clusters on raw data, projected on two first principal components.

for unsupervised clustering (see Figure 9. Now, if we compute the clusters given by the GMM on the entire raw data set, and project this onto the space generated by the first two components, we observe that this clusters can identify the two speakers, profiting from the underlying structure that we could observe in the data. In Figure 14, in green and blue we see the centers of the clusters projected into the first two components.

7 Discussion

Given the novel nature of the problem that we studied, we encountered a fair number of difficulties and foresee many challenges for which we provide some discussion. We start by noting that the special structure we were dealing with not only can be the seed of new applications, but also allowed us to isolate specific aspects of the process of diarization and focus on this setting in particular.

If one were set on using distance-based clustering methods, we believe that hierarchical trees with KL Divergence is the best candidate. We present our data-specific reasons for this choice in Section 3. However, our experiments show that this method will have limited utility in practice. First of all, computing the KL Divergence for all pairs of data points is computationally expensive. Secondly, getting a decent accuracy seems to require substantial tuning or manual data analysis. In Section 3 we present a somewhat successful case, which perfectly shows the limitations of his method in the best of scenarios. In particular, some voices appear indistinguishable, while in order to accurately cluster the distinguishable ones, we had to experiment with the number of clusters we allow in our model. In Figure 5 we illustrate a contrary example where no amount of tuning on the number of clusters gave any significant results.

We recognize that our data set is imperfect – as we mention in Section 2.3, some of our data points actually include multiple speakers. Analyzing the extent of this problem, and the efficiency of hierarchical trees in the absence of this issue is out of scope of this project.

Our supervised method is prompted by the observation that 1. our data might be more separable than cluster-able, and 2. that cluster characteristics may be most evident when looking on all data points at the same time. We believe that with a representative labeled sample, this method could prove very effective, especially since we could solve a classification problem after using PCA to reduce the dimensionality of our data.

As an optimistic remark, we recall the efficacy of GMM on identifying change of speakers. This can be used to complement the clustering model, for example, by first running the change identification and then using that to cluster the voices that were identified as being separate.

Finally, as mentioned in the introduction, more recent approaches include the use of neural networks or supervised learning. We see a future line of work draw these techniques into this problem setting. In particular, we believe that the use of recurrent neural networks can be of help as there is likely to exist some regularity on how many consecutive lines a speaker speaks, and time-state dependent models may be useful in this case.

Acknowledgments

Andrey Sushko wrote code for processing data, generating feature sets for the algorithms, and quantifying and displaying results. Andreea Georgecsu wrote code for k-means, KL divergence, hierarchical tree clustering, and the supervised method. Patricio Foncea wrote code for

Earth-Mover distance, PCA, and Gaussian mixture models. All planning and analysis discussions were done together as a group. We thank Zhi Xu and Suvrit Sra for useful discussion and feedback during the preparation of this project.

References

Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. 2012. Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):356–370.

Dimitrios Dimitriadis and Petr Fousek. 2017. Developing on-line speaker diarization system. In *Proc. Interspeech*, pages 2739–2743.

Daniel Garcia-Romero, David Snyder, Gregory Sell, Daniel Povey, and Alan McCree. 2017. Speaker diarization using deep neural network embeddings. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 4930–4934. IEEE.

Margarita Kotti, Vassiliki Moschou, and Constantine Kotropoulos. 2008. Speaker segmentation and clustering. *Signal processing*, 88(5):1091–1124.

Mohammad H Moattar and Mohammad M Homayounpour. 2012. A review on speaker diarization systems and approaches. *Speech Communication*, 54(10):1065–1103.

Huazhong Ning, Ming Liu, Hao Tang, and Thomas S Huang. 2006. A spectral clustering approach to speaker diarization. In *Ninth International Conference on Spoken Language Processing*.

Gregory Sell and Daniel Garcia-Romero. 2014. Speaker diarization with plda i-vector scoring and unsupervised calibration. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 413–417. IEEE.

Stephen H Shum, Najim Dehak, Réda Dehak, and James R Glass. 2013. Unsupervised methods for speaker diarization: An integrated and iterative approach. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(10):2015–2028.

Sue E Tranter and Douglas A Reynolds. 2006. An overview of automatic speaker diarization systems. *IEEE Transactions on audio, speech, and language processing*, 14(5):1557–1565.

Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. 2018. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4879–4883. IEEE.

Quan Wang, Carlton Downey, Li Wan, Philip Andrew Mansfield, and Ignacio Lopz Moreno. 2018. Speaker diarization with lstm. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5239–5243. IEEE.

Zbynek Zajıc, Marek Hrúz, and Ludek Müller. 2017. Speaker diarization using convolutional neural network for statistics accumulation refinement. *Proceedings Interspeech (2017, in press)*.

Aonan Zhang, Quan Wang, Zhenyao Zhu, John Paisley, and Chong Wang. 2018. Fully supervised speaker diarization. *arXiv preprint arXiv:1810.04719*.