Odyssey 2010
The Speaker and Language Recognition Workshop
28 June – 1 July 2010, Brno, Czech Republic

Speaker linking in large data sets

David A. van Leeuwen

TNO Human Factors, Soesterberg, The Netherlands
Radboud University, Nijmegen, The Netherlands
david.vanleeuwen@tno.nl

Abstract

This paper investigates the task of linking speakers across multiple recordings, which can be accomplished by speaker clustering. Various aspects are considered, such as computational complexity, on/offline approaches, and evaluation measures but also speaker recognition approaches. It has not been the aim of this study to optimize clustering performance, but as an experimental exercise, we perform speaker linking on all '1conv-4w' conversation sides of the NIST-2006 evaluation data set. This set contains 704 speakers in 3835 conversation sides. Using both on-line and off-line algorithms, equal-purity figures of about 86 % are obtained.

1. Introduction

In his book "Blink" Malcolm Gladwell describes how during World War II, British interceptors of Morse-code traffic could gain information from the enemy by recognizing the so-called 'fist' of the operator making the transmission. Although the messages were encrypted, the performance of producing the Morse-code on air is person-dependent, and the characterization of this performance was coined the 'fist' of the operator. By recognizing the operator, and linking broadcasts from different times and places together, movements of the enemy could be inferred. This *linking* is what we consider now as a task for speaker recognition: track unknown speakers along a sequence of recordings.

Linking, also known as clustering, is a task occurring in many other domains, too. For instance, in image processing, detecting faces in images and linking them together is technology available in commercial and online photo organizing tools. Here, the user helps out the face clustering algorithm by explicitly including or excluding newly found faces in the photo library. The fact that faces typically receive an identity from the user is irrelevant to the clustering algorithm: the images are only compared within the data collection. These face recognition implementations appear to work quite well from a user perspective. The technology might be called semi-supervised, as the user's input is required iteratively in the entire clustering process. For speaker linking a semi-supervised clustering implementation is still quite a challenge, because the linear time aspect of speech is more difficult to deal with in presenting candidate samples to the user than the 2-dimensional capabilities of the visual interface.

Speaker linking is related to speaker diarization, which has an additional task of first segmenting an audio stream in segments of a single speaker (ignoring overlap for the moment) and then clustering the segments in order to find which segments belong to the same speaker, answering the question who spoke when. Speaker diarization is typically performed within a single recording, whereas speaker linking explicitly is about a (possibly large) set of separate recordings. In speaker linking the segmentation is assumed to be found by out-of-band information, e.g., the signal strength of a radio or the granularity of phone communication.

In some way speaker linking occurs in NIST Speaker Recognition Evaluation (SRE) tasks of "2-wire" training, where the common speaker in three conversations must be found for building a proper speaker model. In this task an additional diarization tasks exists because the recordings are made with both speakers of the conversation on the same channel. However, in that specific task, there are typically three conversations that must be linked, whereas in speaker linking we aim at hundreds or thousands of speakers.

Another task that is closely related to speaker linking is that of speaker tracking. Here the task is to find spoken segments of a particular speaker for which some training material is given. This has both aspects of diarization, as the audio stream needs to be segmented first, and of operation on a collection of recordings. It has as such been performed in both the 1999/2000 editions of the NIST SRE for 2-wire telephone conversations and the French ESTER 2005 evaluation campaign for broadcast news. The task and evaluation measures were cast in a detection and retrieval framework, respectively, and can be compared to a "known item retrieval" task in text retrieval. In speaker linking, the fact that there are no reference speakers implies that all segments need to be related to all others, which makes the task more complex and sensitive to errors.

As a final relation to speaker linking there obviously is the problem of clustering in general. In many machine learning and pattern recognition problems there is the challenge to cluster items in an unsupervised way, where the number of classes is unknown. Examples are clustering of text documents or images. The way in which speaker linking differs from general clustering is the way the classes are defined: these are the actual speaker identities, and there is in principle no doubt about the ground truth in class labels, which is different from, e.g., the case of topic in text document clustering.

This paper is organized as follows. First, we make an inventory of the task and the challenges in speaker linking, and consider some approaches to the problem. Then we will characterize the various task aspects, and in the experimental section apply two clustering approaches to a fairly challenging data set from the NIST 2006 SRE.

2. Task and Aspects of Speaker Linking

In this paper, we use the term *linking* for the task and *clustering* for the method or algorithm to permorm this task.

The task of speaker linking can be defined as follows.

> Given a collection of speech segments $\mathcal{S}$, link segments originating from the same speaker together.

The number of speakers S is unknown, and can range from 1 to the number of segments N. The number of possible linking configurations is a combinatorial problem. The number of possible partitions of the segments in non-empty clusters is known as the Bell-number [1] B_N, which grows very fast with N.[1] For this reason, any clustering approach that investigates all possible linking configurations, and evaluates the likelihood of each, is intractable. Tractable algorithms therefore use iterative procedures that make heuristic assumptions that a previous clustering step provides a good prior to a subsequent clustering step in terms of the optimal linking configuration. We will now look at some aspects of clustering.

2.1. Some clustering approaches

For a general introduction to clustering approaches, see [2]. Of the iterative clustering procedures, the broadest separation is in *top down* (divisive) or *bottom-up* (agglomerative). In speaker diarization, agglomerative clustering has been the dominant approach in recent years [3], but also divisive schemes have been studied [4]. In agglomerative clustering, clusters are initialized with the original segments, and iteratively the two clusters are found that are the best candidates for being merged. Alternatively, in divisive clustering a single cluster is initialized with all segments, and iteratively the best partitioning into more clusters is found. In both approaches, merging/partitioning ends when some stopping criterion is met.

A fairly careful approach to agglomerative clustering starts with $C = N$ clusters and reduces the number of clusters to $C - 1$ by considering all possible $\frac{1}{2}C(C-1)$ cluster pairs for a merge, and iterating. This procedure runs in $\mathcal{O}(N^3)$ time. A faster approach considers a number of cluster pairs simultaneously (hierarchical clustering), reducing computation time to typically $\mathcal{O}(N^2 \log N)$.

In divisive clustering the number of possible partitionings to consider at every iteration is combinatorically large, again. By making use of a model for a cluster, and splitting a cluster according to the model parameters, the likelihood of the segments given the (new) models can be used to efficiently partition the segments.

2.2. On-line vs off-line clustering

In many application areas, the data (speech segments in our case) are collected sequentially. Often, it is desirable to have a linking available at any instant in time, or at least periodically. For the typical agglomerative clustering described above the order in which the segments arrive does not play a role. We call this off-line, or after the fact, clustering. If one needs to re-cluster the entire collection with the advent of each new segment, the computation time necessary scales with $O(N^4)$—still, at each individual step with $\mathcal{O}(N^3)$—indicating that sooner or later there won't be not enough time to complete a full agglomerative clustering.

An alternative approach is *on-line* clustering, where we keep the C_N clusters found after N segments. We only consider assigning a new segment to an existing cluster or creating a new cluster for this segment. At segment instant N, the computational requirements then scale only as $\mathcal{O}(C_N)$ which much less quickly exceeds computing power. As already noticed in the context of speaker change detection and clustering using a Bayesian Information Criterion, on-line clustering may in practice perform just as well, or even better, than its full agglomerative counterpart [5].

2.3. Re-training of cluster models

One thing to consider during speaker clustering is whether or not to re-train speaker models for the clusters. For divisive clustering, this typically happens because some form of modeling of the entire cluster is necessary to make the partitioning tractable. For agglomerative clustering, it is possible to work with one speaker model for each segment, thereby only clustering in a logical sense. The alternative is to re-train a new model after each cluster merge, with obvious additional computational costs but a possibly better speaker model.

[1] According to the On-Line Encyclopedia of Integer Sequences, the sequence B_N starts conservatively as 1, 1, 2, 5, 15, 52, ..., but already $B_{20} \approx 5.2 \times 10^{10}$

In speaker diarization, it is normal to re-train cluster models after a merge; in fact, for every possible merge under consideration (typically $\frac{1}{2}C(C-1)$) a new model is trained. For the GMMs (often used in diarization), this is computationally the most expensive step, thereby limiting the duration of the audio that can be processed.

For some modeling techniques training a new model from a new combination of speech segments can be computed efficiently. An example is the dot-scoring approach for UBM-GMM models. Here, the GMM representing the speaker model is seen as a perturbation of the UBM, using the linear term of the Taylor-series expansion. The speaker model is built from a simple division using first and zeroth order Baum-Welch statistics of the training data given the UBM, so if the statistics per segment are kept in memory, building a new model using a new cluster configuration can be carried out very efficiently. Similarly, in GMM-SVM modeling the Gram-matrix of the background and all known segments can be kept in memory, and only the kernel function of the existing set of segments with the new segment needs to be evaluated before a new SVM model can be computed.

2.4. Within-set normalization

In NIST SRE it is, for very good reasons, not allowed to use information of segments in the evaluation other than the segments used for training and testing in the trial under consideration. For the task of speaker linking, however, it is not unreasonable to allow for the use of all available segments for any decision regarding two (or more) segments. Typical usage of available segments could be score normalization or discriminative training. E.g., one might want to utilize known non-target segments for either discriminative training or T- or Z-normalization—but knowing whether or not segments are linked to the same speaker or not is part of the task, so care must be taken. One could consider, for instance, using only segments with very low speaker recognition scores for a particular segment.

2.5. Hard versus soft clustering

So far we have implicitly assumed algorithms that make a hard many-to-one decision as to which segment belongs to which cluster. From experience in many areas in machine learning we may expect that a 'soft' attribution of a segment to a cluster may be advantageous during the clustering process. E.g., in GMM training each data point x is 'distributed' over existing mixtures j and its posterior probability $P(j|x)$ governs the statistics of the next iteration mixtures.

3. Evaluation measures

3.1. Review of existing measures

There are several evaluation measures available for clustering tasks. In speaker diarization, the standard performance measure is the Diarization Error Rate [6] (DER). This is a time-weighted error measure, accounting for wrongly attributed speaker, missed speech and false alarm speech. Like in speaker linking, in speaker diarization speakers are labeled with absolute identities, therefore an optimal mapping of hypothesized speakers to reference speakers is made that minimizes the DER—similar to word alignment in determining the word error rate for a speech recognition system. The DER is expressed as a fraction of speech, and the denominator accounts for all speech of all speakers, including overlap. For accurate and meaningful determination of the DER, forced-aligned reference speech is necessary [3]. The fraction of false alarm speech, therefore, has an unusual interpretation as it is not normalized by the amount of non-speech but by the amount of speech in the audio [7].

In the NIST SRE 2000 speaker tracking task, performance was measured in a time-weighted version of Miss and False Alarm rates. Here hypothesis segments of speech were scored for target model speakers.

In the ESTER 2005 speaker tracking task retrieval measures were computed: time-weighted versions of recall and precision, and the final evaluation measure was the F-measure, the harmonic mean of recall and precision.

The measures mentioned above are all time-weighted, and are tailored to include the diarization aspects of the task at hand. Because in our task definition of speaker linking the speaker segmentation is not part of the problem, we would like to concentrate on the clustering aspects. We defined two *impurities*, of cluster and speaker. The former measures to what extent a cluster contains segments from different speakers [2], the latter measures to what extent reference speakers are distributed among clusters.

3.2. Cluster impurity and entropy

Consider the found clusters $\mathcal{C}_i$, $i = 1, \ldots, C$, each a set of one or more segments j. Then define $R(j)$ the true (or reference) speaker of segment j, and $f_k(R(\mathcal{C}_i))$ the frequency of reference speaker k occurring in $\mathcal{C}_i$, where $k = 1, \ldots, K_i$ and reference speakers are ordered in decreasing frequency within $\mathcal{C}_i$. Using the short-hand

$$f_{ik} = f_k(R(\mathcal{C}_i)), \tag{1}$$

the number of segments in cluster i is $n_i = \sum_k f_{ik}$, and the total number of segments is $N = \sum_i n_i$. Then the *cluster impurity* I^c is defined as

$$I^c = 1 - \frac{1}{N} \sum_i f_{i1} \tag{2}$$

Note that this is a segment-weighted average of individual cluster impurities $1 - f_{i1}/n_i$. Another measure that is used the evaluation of cluster purity is the cluster entropy,

for cluster i defined using the probability $p_{ik} = f_{ik}/n_i$ as

$$H_i^c = -\sum_k p_{ik} \log_2 p_{ik}. \quad (3)$$

Again, we can average this entropy over the entire collection of clusters, weighting by segment

$$H^c = \frac{1}{N} \sum_i n_i H_i^c. \quad (4)$$

The speaker impurity ranges from 0 to (almost) 1, where the lowest value indicates perfect separation of speakers. The entropy is non-negative, where 0 indicates that there is no doubt about what speakers are assigned to a cluster.

3.3. Speaker impurity and entropy

It is trivial to obtain cluster impurity and entropy 0 by assigning each segment to its own cluster. We must therefore also look at the other side of the task—the ability to link segments of the same speaker together. Starting this time from the reference speakers k consider the set of segments $\mathcal{S}_k$ spoken by k. For each segment $j \in \mathcal{S}_k$, determine the cluster $C(j)$ to which is was assigned, and compute the frequencies $g_{ki} = g_i(C(\mathcal{S}_k))$ of assigning speaker k to the same cluster, in descending order. We have $m_k = \sum_i g_{ki}$, the number of segments spoken by speaker k, and $N = \sum_k m_k$, the total number of segments. The *speaker impurity* I^s is defined as

$$I^s = 1 - \frac{1}{N} \sum_k g_{k1}. \quad (5)$$

Similarly, we can set $q_{ki} = g_{ki}/m_k$ and define the *speaker entropy* for speaker k as

$$H_k^s = -\sum_i q_{ki} \log_2 q_{ki}, \quad (6)$$

and, averaging over all speakers, weighting by the amount of segments per speaker, we obtain

$$H^s = \frac{1}{N} \sum_k m_k H_k^s. \quad (7)$$

Note that, again, there is a trivial solution with $I^s = 0$ and $H^s = 0$ by putting all segments in a single cluster.

3.4. Trade-off analysis

Similar to speaker detection, in speaker linking we have two trivial solutions that result in either $I^c = 0$ or $I^s = 0$. It therefore makes sense to analyze results in terms of a trade-off between speaker and cluster purity (or entropy). The clustering algorithm is likely to have some stopping criterion, which has a parameter (e.g., a threshold) that governs the trade-off. In the tradition of speaker detection, we can make a parametric plot the trade-off. For the impurities, which have the dimension of a fraction, we can do this on probit-warped axes, similar to the infamous DET-plots that we all have learned to like so much in the past 15 years.

4. Linking experiments

4.1. Speech data

In this section we report some experiments carried out on a medium-sized collection of speech segments. Since this work was carried out during the preparation for NIST-2008, we have used the NIST-2006 SRE speech material for experiments. We used all 3835 test segments available in the '1conv4w' test condition, which contains both male and female speech segments. We used a fixed, but otherwise random, order of these segments in order to simulate 'on-line' experiments.

4.2. Speaker recognition system

We used a snapshot of the TNO speaker recognition system under development for the NIST SRE-2008 evaluation. Specifically, we used a GMM-SVM system, state-of-the-art for SRE 2006 [8] and still competitive in 2008, with NAP [9] and T-norm [10]. Twelve PLP features plus log energy, augmented with derivatives were extracted, energy-based speech activity detection was applied and zero mean, unity variance normalized per segment. The 512-mixture UBM was trained gender-independently using 2257 speakers from LDC Switchboard II phase 2 and Fisher English, NIST SRE 2001, 2002, 2003 and 2005, and non-English segments from NIST LRE-2003. With the UBM as prior, point estimates of Gaussian means were made using MAP adaptation [11] for each speech segment. Channel compensation was implemented through NAP, using the NIST SRE 2004 '1side4w-1side4w' data. Shifts in means were stacked in supervectors, and SVM models were built using a background of 1640 speakers included in the UBM. Compacting the (linear kernel) SVM gave the opportunity to a fast (inner-product) way of scoring. Scores were T-normalized using 155 speakers from NIST SRE-2004.

This system, when evaluated on NIST SRE-2006 data, had a performance of 5.06 % EER. For subsequent experiments, we computed a full matrix of all test segments scored against all models built from the same test segments. The matrix was made symmetric by averaging with its transpose.

4.3. On-line algorithm

In order to simulate an 'on-line' experiment we used the information from score matrix S_{ij} (score of segment j for model i) in the following way. A cluster $\mathcal{C}$ consists of a list of segment indexes $\{i\}$ that belong to that cluster.

1. The first segment $j = 0$ is assigned a new cluster $\mathcal{C}_0$, and set the number of clusters $C = 1$.

2. Continue with the next segment j, determine the maximum score $m_j = \max_{i=0}^{j-1} S_{ij}$ and the model index $i_{\max}$ of the maximum

Table 1: Cluster and speaker imprecision and entropy for the on-line clustering approach using the NIST SRE 2006 test data. Also indicated are the threshold θ and the number of speakers found C (true number is 704).

θ	I^c	I^s	H^c	H^s	C
2.0	0.926	0.058	5.993	0.173	22
2.5	0.827	0.076	4.330	0.224	63
3.0	0.741	0.081	3.662	0.240	115
3.5	0.628	0.084	2.863	0.251	193
4.0	0.494	0.093	2.107	0.274	316
4.5	0.340	0.104	1.276	0.308	473
5.0	0.232	0.115	0.805	0.345	632
5.5	0.151	0.142	0.497	0.424	779
6.0	0.102	0.164	0.323	0.494	917
6.5	0.061	0.190	0.179	0.577	1040
7.0	0.044	0.227	0.124	0.701	1167
7.5	0.028	0.260	0.076	0.821	1309
8.0	0.022	0.289	0.059	0.935	1443
8.5	0.017	0.331	0.046	1.082	1592
9.0	0.014	0.381	0.039	1.258	1768
9.5	0.013	0.419	0.034	1.393	1911
10.0	0.011	0.456	0.029	1.542	2096

3. If $m_j > \theta$, assign segment j to the cluster to which the segment $i_{\max}$ belongs; otherwise create a new cluster $\mathcal{C}_C$, assign segment j to this cluster, and increment the cluster count C.

4. Continue with step 2 until all segments are assigned to clusters.

This on-line algorithm represents a cluster as a collection of segments and considers a new segment by comparing it to all the segments of the cluster. Clusters compete for the new segment (this is similar to identification) because only the highest scoring model segment can claim the new segment for its cluster, and hence there are hard decisions made as to which segment belongs to which cluster. The threshold parameter θ governs the trade-off between clusters containing more than one speaker and speakers distributed over more than one cluster.

In Table 1 the performance results for a scan of thresholds in the range 2–10 (in "units" of T-norm scores) is shown. At $\theta = 5.56$ "equal impurity" of 0.145 is reached, and coincidentally around the same value the correct number of speakers (704) is crossed. Note that in such an on-line implementation it is not trivial to use information of the number of speaker in the collection, and sweeping the threshold after-the-fact, as we are used to in producing DET-plots, does not have a natural parallel in on-line clustering.

4.4. Off-line algorithm

From the score matrix S_{ij} we can also simulate an off-line clustering algorithm along similar lines.

Table 2: Cluster and speaker imprecision and entropy for the off-line clustering approach using the same NIST SRE 2006 score matrix shown in Table 1. Note that the equal impurity point is crossed at a different threshold.

θ	I^c	I^s	H^c	H^s	C
5.0	0.908	0.008	8.159	0.027	139
5.5	0.775	0.022	6.563	0.073	281
6.0	0.605	0.039	4.791	0.128	443
6.5	0.446	0.062	2.803	0.201	607
7.0	0.286	0.091	1.341	0.300	783
7.5	0.146	0.133	0.552	0.442	1000
8.0	0.088	0.180	0.271	0.592	1172
8.5	0.044	0.231	0.128	0.769	1369
9.0	0.027	0.271	0.079	0.922	1521
9.5	0.022	0.311	0.062	1.069	1684
10.0	0.015	0.370	0.042	1.269	1882
10.5	0.011	0.419	0.029	1.449	2083
11.0	0.009	0.459	0.023	1.598	2267
11.5	0.008	0.501	0.021	1.745	2449
12.0	0.007	0.541	0.018	1.884	2614

1. Initialize $C = N$ and set each segment i to cluster $\mathcal{C}_i$.

2. find the highest score $s_{\max}$ in the upper-triangular matrix $s_{i<j}$, and the corresponding row i and column j

3. if $s_{\max} < \theta$, stop.

4. merge the segments in clusters $\mathcal{C}_i$ and $\mathcal{C}_j$, and remove column j from the matrix

This agglomerative clustering algorithm is comparable to the on-line algorithm in the sense that it does not re-train models after merging clusters; merging is based entirely on individual segment-segment scores. Again, model segments are competitive as from all scores in a column only the maximum survives, other (possibly high) scores for that segment are not considered further.

In Table 2 the results for the off-line algorithm are given using exactly the same score matrix S_{ij} as used in the on-line algorithm. Compared to the on-line clustering, the "calibration" of the threshold is quite different, the equal impurity point of 0.149 is reached at $\theta = 7.75$, where the correct number of speakers is found for θ just below 7. Note that here, as an efficiency step, we could have dumped the cluster status at regular intervals of the merging score during the agglomerative clustering, thus scanning a range of thresholds in a single clustering operation. This is similar to computing DET statistics by sorting all scores in a speaker recognition evaluation.

4.5. Effect of collection size

One of the challenges of speaker linking is that the problem gets harder if the number of speakers and segments

Table 3: Performance of the off-line clustering algorithm with smaller sample size. Performance is summarized in the Equal Impurity I and the threshold θ at which this is reached. Also, the number of speakers retained S and samples N is shown, as well as the average number of segments per speaker.

fraction	θ	I	S	N	N/S
1	7.56	0.139	704	3835	5.45
1/2	6.79	0.123	607	1932	3.18
1/4	6.32	0.103	470	940	2.00
1/8	6.0	0.083	323	482	1.49
1/16	5.9	0.064	195	236	1.21

increases. In order to study this experimentally, we ran the off-line clustering system on the collection reduced in size. Using different fractions, we took a random sample from the 3835 segments and ran the clustering. The results are shown in Table 3. Clearly, performance increases as the number of speakers and segments decreases.

5. Discussion

The impurity columns in Table 1 and 2 are plotted together in a trade-off plot, using common DET scales, in Figure 1. Interestingly, the performance is very similar, with the off-line clustering somewhat better in the low speaker impurity range. The trade-off graphs are not as straight as we are used to in speaker recognition DETs, but we must realize that this is the result of quite a different operation. There is a relation between false alarms and impure clusters, since high-scoring impostor segment comparisons will lead to mixed clusters. Similarly there is a relation between impure speakers and misses.

If impurities were to be compared to errors, then we can see that the complexity of clustering increases the error rates with respect to speaker detection: for the full NIST test set, the equal error rate in detection is 5.06 % for this system, but the equal impurity is 13.9 %. Clustering is inherently a more difficult task, because detection errors lead to impure clusters, which will provoke more errors.

If the problem is made simpler, by selecting a random subset of the segments as we have shown in Table 3, the clustering performance increases again. This can be due to the fact that as a consequence of the sampling, the number of segments per speaker decreases, and hence a trivial solution of giving every segment its own cluster becomes better performing.

We have discussed a number of characteristics of clustering, but have performed experiments with only two approaches, an on-line and an off-line algorithm. It is remarkable that the on-line and off-line algorithms perform

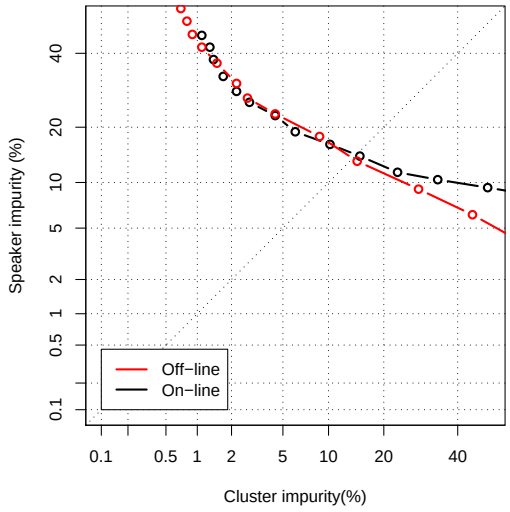

Figure 1: Trade-off between cluster and speaker impurity, for both the on-line and off-line algorithm. Scales of the axes are warped by the probit function, as is customary for DET plots in speaker recognition.

quite similar, where we would expect the off-line agglomerative clustering to have more information available and make better decisions. However, in the context of speaker diarization a similar effect has been seen before in literature [5].

The algorithms were pretty straightforward—it was not the aim of this research to develop better clustering algorithms but rather to investigate some of the issues that arise in speaker linking, especially when collection sizes increase. It would be interesting to investigate soft-clustering algorithms, and see if they are less susceptible to the problem that clusters become impure when errors are made.

As a final note, we became aware of another paper submitted to this conference "The Speaker Partitioning Problem" [1] after the original submission deadline, and the authors treat the exact same task, but have a more mathematical approach to the problem. They show, for instance, that many tasks in speaker recognition can be seen as special cases of speaker partitioning (or linking, as we have termed it). Our focus is more on problems with a high number of speakers and segments, and the scalability issues that are related to that. Where in speaker detection performance stays constant with more segments or speakers, this is not the case for speaker linking. We believe that there are still a lot of interesting issues to be investigated, including re-training, within-collection discriminative training and normalization, soft clustering and scalability.

6. References

[1] Niko Brümmer and Edward de Villiers, "The speaker partitioning problem," in *Proc. Odyssey 2010: The Speaker and Language Recognition Workshop*, Brno, June 2010, Submitted.

[2] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar, *Introduction to Data Mining*, chapter Chapter 8. Cluster Analysis: Basic concepts and algorithms, Adison-Wesley, 2006, ISBN: 9780321321367.

[3] Xavier Anguera, Chuck Wooters, and Jose M. Pardo, "Robust speaker diarization for meetings: Icsi rt06s meetings evaluation system," in *Machine Learning for Multimodal Interaction*, vol. 4299 of *Lecture Notes in Computer Science*, pp. 346–358. Springer Berlin/Heidelberg, 2006.

[4] Sylvain Meignier, Jean-François Bonastre, and Stephane Igounet, "E-hmm approach for learning and adapting sound models for speaker indexing," in *Proc. Odyssey 2001: The Speaker Recognition Workshop*, 2001, pp. 175–180.

[5] Alain Tritschler and Ramesh Gopinath, "Improved speaker segmentation and segments clustering using the Baysian Information Criterion," in *Proc. Eurospeech*, 1999.

[6] Jonathan G. Fiscus, Jerome Ajot, and John S. Garofolo, "The rich transcription 2007 meeting recognition evaluation," in *The Joint Proceedings of the CLEAR 2007 and RT 2007 Evaluation Workshops*, vol. 4625 of *LNCS*, pp. 373–389. Springer, 2007.

[7] David A. van Leeuwen and Marijn Huijbregts, "The AMI speaker diarization system for NIST RT06s meeting data," in *Machine Learning for Multimodal Interaction*, vol. 4299 of *Lecture Notes in Computer Science*, pp. 371–384. Springer Berlin/Heidelberg, 2006.

[8] Niko Brümmer, Lukáš Burget, Jan Černocký, Ondřey Glembek, František Grezl, Martin Karafiát, Pavel Matějka, David A. van Leeuwen, Petr Schwarz, and Albert Strassheim, "Fusion of heterogeneous speaker recognition systems in the STBU submission for the nist speaker recognition evaluation 2006," *IEEE Transactions on Speech, Audio and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.

[9] William Campbell, Douglas Sturim, Douglas Reynolds, and Alex Solomonoff, "SVM based speaker verification using a GMM supervector kernel and NAP variability compensation," in *Proc. ICASSP*, Toulouse, 2006, IEEE, pp. 97–100.

[10] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, "Score normalization for text-independent speaker verification systems," *Digital Signal Processing*, vol. 10, pp. 42–54, 2000.

[11] J.-L. Gauvain and C.-H. Lee, "Maximum a posteriori esitimation for multivariate gaussian mixture observations of markov chains," *IEEE Trans. Speech Audio Processing*, vol. 2, pp. 291–298, 1994.