The AMI speaker diarization system for NIST RT06s meeting data

David A. van Leeuwen[1] and Marijn Huijbregts[2]

[1] TNO Human Factors
Postbus 23, 3769 Soesterberg, The Netherlands
david.vanleeuwen@tno.nl
[2] Department of EEMCS, Human Media Interaction
University of Twente, Enschede, The Netherlands
marijn.huijbregts@utwente.nl

Abstract. We describe the systems submitted to the NIST RT06s evaluation for the Speech Activity Detection (SAD) and Speaker Diarization (SPKR) tasks. For speech activity detection, a new analysis methodology is presented that generalizes the Detection Erorr Tradeoff analysis commonly used in speaker detection tasks. The speaker diarization systems are based on the TNO and ICSI system submitted for RT05s. For the conference room evaluation Single Distant Microphone condition, the SAD results perform well at 4.23 % error rate, and the 'HMM-BIC' SPKR results perform competatively at an error rate of 37.2 % including overlapping speech.

1 Introduction

In the ongoing series of evaluations of speech technology the task of transcription and speaker diarization of meeting data has received renewed interest of several research groups, judged from the number of participants to the NIST Rich Transcription evaluation in Spring 2006 (RT06s) [1]. This paper describes the technologies employed in the Speaker Diarization (SPKR) and Speech Activity Detection (SAD) tasks by the AMI (Augmented Multiparty Interaction) team. This submission was the continuation of the RT05s work carried out by TNO [2], and was this year augmented by co-workers from the University of Twente. Further, in co-operation with AMI partners from the University of Edinburgh [3] an attempt was made to actually use the information of multiple microphones.

The task of Speaker Diarization in meetings can be summarized as a task to find out by automatic means 'who spoke when,' given the simultaneous recordings of several distant microphones located in the room. New challenges in the 2006 edition of RT included the scoring of overlapped regions of speech.

The AMI team tried to improve last year's approach by extending the BIC segmentation/clustering framework with Viterbi re-segmentation, and by exploring new clustering methods based on the work carried out by ICSI [4]. Further,

some attempts were made to create systems that could produce overlapping speakers in the output.

During the workshop held to discuss the results of RT06s it turned out again how important it is to do speech activity detection properly before processing the microphone signal for further analysis (speaker diarization and speech-to-text). Although we employed a very basic SAD system, the performance was good for meeting room data, and we will therefore start the paper describing the SAD system, with training data bases and results. We will then continue by describing the three speaker diarization systems submitted to RT06s.

2 Speech Activity Detection

As pointed out earlier [2] the SAD task is implicitly important to the SPKR task due to the evaluation measure used in the SPKR task. In short, any period of the analysis that is mis-classified as either missed speech (miss) or, especially, false-alarm speech (FA) can only add to the speaker Diarization Error Rate (DER). Unless the SPKR system tries to deal with SAD itself, a badly performing SAD can be detrimental to the DER, even if the SPKR system itself functions very well.

The SAD system used here is a slightly improved version described for RT05s [2]. We use two Gaussian Mixture Models (GMMs) to model speech and silence, and use a Viterbi decoder to find the state sequence that maximizes the likelihood of the model sequence, given the data.

2.1 Databases

For the development of the SAD (and SPKR) systems, we exclusively used the RT05s evaluation meeting room data, henceforth named 'devtest' data. Only for the SAD system we used training material, for which we chose the 10 annotated AMI meetings distributed for RT05s development testing.

2.2 Microphone conditions

Two sets of microphone conditions were investigated:

SDM Single distant microphone. We used the microphone, as designated by NIST, without further pre-processing.

Array-beam Multiple distant microphones with beam forming. For this condition we used the output of a delay-and-sum beam-former built by co-workers within the AMI project [3]. We used the output of the beam-formed acoustic signal as a single channel for further processing.

Note, that because our SDM initially gave us better devtest results than the beam-formed MDM data, we submitted both systems as (compulsory) 'MDM' condition, and designated our SDM as 'primary MDM.'

2.3 Features, models, decoder

This year we used RASTA-PLP features as extracted by ICSI's `rasta` tool [5] developed within the SPRACH project. Twelve PLP coefficients plus log energy were extracted over a window of 32 ms at a frame rated of 16 ms. First order derivatives of the 13 features were determined over 7 consecutive frames. The 26 features were modeled in a GMM with 16 components, trained with the 10 annotated AMI meetings. Two GMMs were thus trained, and a 2-state HMM network was built. Rather than training the transition probabilities $a(j|i)$, we took fixed values by setting:

$$\begin{gathered} \frac{a(\text{speech}|\text{sil})}{a(\text{sil}|\text{speech})} = R \\ a(\text{speech}|\text{sil}) + a(\text{sil}|\text{speech}) = P \\ \textstyle\sum_j a(j|i) = 1 \end{gathered} \tag{1}$$

and choosing $\log P = -30$ and $R = 10$. The parameters P and R were chosen to minimize SAD error for devtest data.

Last year's SAD system contained an error in the backtracking code of the Viterbi decoder, which had to be patched with a boxcar filter to smooth out state transitions. After removing this 'bug' from the system, post-filtering of the state sequence was no longer necessary. The only fenomenological filtering that we still applied was the removal of speech segments of duration shorter than 0.5 s, a filter that might not have been optimal if the SAD and SPKR error measures did not have defined a 'forgiveness' collar of 0.25 s around speech segments.

2.4 SAD performance

In Table 1 we summarize the SAD results in terms of the Speech activity detection Error Rate on devtest and the RT06s evaluation data, the latter for both meeting room data and lecture room data. The SAD error rate is defined as

$$e_{\text{SAD}} = e_{\text{FA}} + e_{\text{miss}} = \frac{T_{\text{FA}} + T_{\text{miss}}}{T_{\text{speech}}} \tag{2}$$

where T_{FA}, T_{miss} and T_{speech} are aggregated times of silence misclassified as speech, speech misclassified as silence, and total duration of speech under evaluation.

A few observations can be made from Table 1. First, the single central microphone condition appears to outperform the array beam-formed signal processing slightly. This may seem due to the fact that the GMM models have been trained on SDM data only, but we have experimented with models trained on array beam-formed data as well, and could not obtain better results than the 3.43 % SAD error shown above. A second remark is that the performance for the Lecture room data is much worse than for the meeting room data, which will have its repercussions to the Speaker Diarization processing of the lecture room data that we present later. This is probably due to the out-of-domain model training,

Table 1. AMI SAD results for devtest and evaluation data, separated for SDM and MDM-array processing. Two sets of FA and miss values are given. The left set are time-weighted values where the *speech* evaluation time is in the denominator, the right set is determined using *non-speech* and *speech* evaluation time, respectively.

		(%)	prior (%)		no prior (%)	
Test set	Microphone	e_{SAD}	e_{FA}	e_{miss}	p_{FA}	p_{miss}
RT05s (devtest)	SDM	2.86	1.5	1.4	31.4	1.4
RT06s conference room	SDM	4.23	2.0	2.2	34.9	2.2
RT06s lecture room	SDM	26.0	6.3	19.7	45.7	19.7
RT05s (devtest)	Array-beam	3.41	2.6	0.8	54.1	0.8
RT06s conference room	Array-beam	4.82	3.8	1.0	66.3	1.0
RT06s lecture room	Array-beam	30.4	8.5	21.9	61.1	21.9

but it is rather worrying to observe that such an apparently simple task as SAD in our implementation is so sensitive to the acoustic domain it is applied to.

A more interesting observation we would want to make here, is the imbalance of FA and miss rates when expressed as detection tradeoff measures $p_{\mathrm{FA}} = T_{\mathrm{FA}}/T_{\mathrm{sil}}$ and $p_{\mathrm{miss}} = T_{\mathrm{miss}}/T_{\mathrm{speech}}$, where T_{sil} is the actual duration of silence in the evaluation. At the workshop following the RT06s evaluation, it was noted that within the CHIL project, the evaluation measures SDER/NDER are defined as p_{FA} and p_{miss} [6]. These measures are insensitive of the evaluation prior $p(\mathrm{speech})$, and in a sense provide better indication of the discriminability of the detector than e_{SAD}. The latter measure is, for a given detector, dependent on the prior $p(\mathrm{speech})$. It is questionable to optimize a detector using an evaluation measure that includes the prior by minimizing e_{SAD}, as we did by choosing a particular value for R.

In speaker detection, which is also in essence a two-class discrimination problem, it is customary to plot the tradeoff between p_{FA} and p_{miss} in a Detection Error Tradeoff plot [7], a ROC-curve on axes scaled by the probit function [8]. The basis of a DET-analysis is formed by two sets of trials (*target* and *non-target*) with *scores* indicating the support for a trial being a target-trial. The fact that a score is used means that the detector does not make actual *decision* for this analysis, and the decisions can be postponed to the moment of plotting (every point on a DET curve corresponds to a given threshold). We would like to generalize this concept to a time-based segmentation where, by segmentation, decisions *have* been made.

2.5 Time-weighted DET curve

We require our detector to not only produce a segmentation in time, but to also give a *likelihood score* for each segment that that segment is speech (the target class). In comparing the hypothesis segmentation with the reference segmentation we can obtain a higher-grained 'evaluation segmentation' of segments that are either correct speech, correct silence, FA, or miss segments. These segments can now be seen as trials, either belonging to target (speech) or non-target (si-

lence) class, depending on the reference segmentation, each with a score. Let T_i be the durations of these evaluation segments, and the scores be s_i, then for obtaining a point on the DET curve we can set a threshold θ, from which we define the FA probability

$$p_{\mathrm{FA}}(\theta) = \sum_{i \in \mathcal{S}_{\mathrm{non}}, s_i > \theta} T_i \Big/ \sum_{i \in \mathcal{S}_{\mathrm{non}}} T_i, \tag{3}$$

and correspondingly the miss probability

$$p_{\mathrm{miss}}(\theta) = \sum_{i \in \mathcal{S}_{\mathrm{target}}, s_i < \theta} T_i \Big/ \sum_{i \in \mathcal{S}_{\mathrm{target}}} T_i. \tag{4}$$

Here $\mathcal{S}_{\mathrm{non,target}}$ denote the set of evaluation segments that are actually non-target and target class, respectively, according to the reference. By varying θ, the time weighted DET-plot $\big(\mathrm{probit}\, p_{\mathrm{FA}}(\theta), \mathrm{probit}\, p_{\mathrm{miss}}(\theta)\big)$ can be constructed. As for traditional DET plot, this process can be calculated efficiently by sorting segments by score and computing FA and miss time cumulatively.

In order to investigate the feasibility of this new analysis, we generated scores by averaging the log likelihood ratios of the speech and silence models over the segments found by the decoder. In Fig. 1 the corresponding time-weighted DET plot is shown for the RT06s meeting room data. The two curves show different settings for the parameter R from Eq. 2. The solid line shows the curve for the submitted parameter setting $R = 10$ that minimized e_{SAD} for devtest data, the dashed line is an contrastive setting $R = 1$. Both curves show a little 'knee bend' around the operating point that corresponds to the *actual* decisions that were made in the segmentation process. The dashed curve shows better general detection capabilities, but happens to perform slightly worse around the optimum operating point defined by the prior p(speech).

3 Speaker diarization systems

For the speaker diarization task we submitted three bottom-up segmentation/ clustering systems. The first was based on a single consecutive BIC segmentation and clustering, the remaining two on iterative Viterbi segmentation and clustering. All system start with the output from the SAD systems described above.

3.1 BIC based segmentation and clustering

The first speaker diarization system is a continuation of the TNO system in RT05s [2]. The Bayesian Information Criterion (BIC) is used to first segment the speech in speaker-homogeneous segments, which are then later clustered based on a Gish distance measure and using a BIC stop criterion. An addition this year was a final Viterbi re-segmentation based on 16-component GMM models that are trained on the clusters found in the BIC clustering process. In this

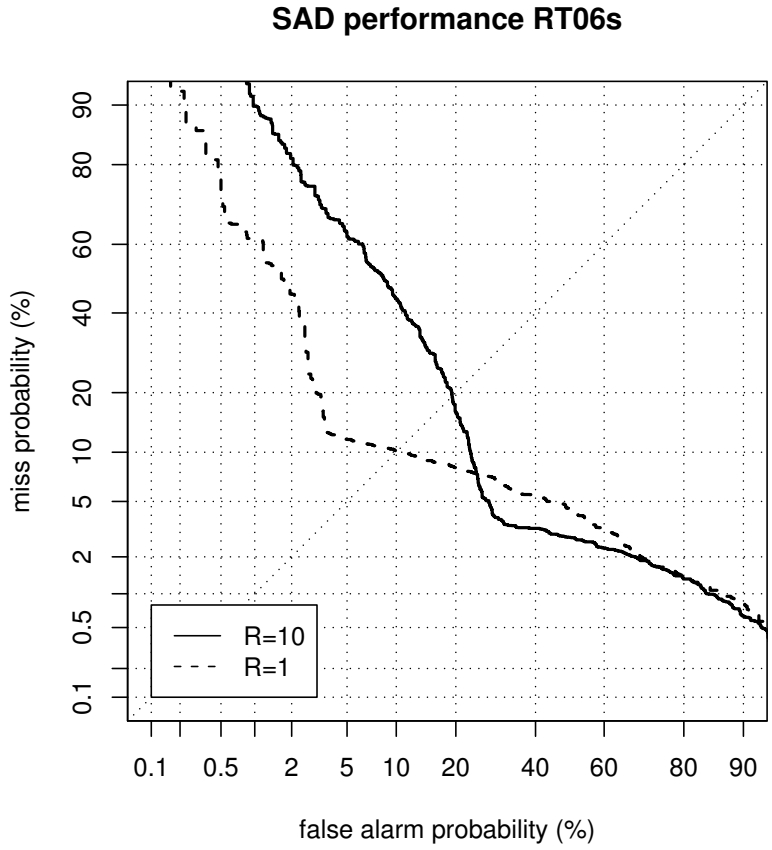

Fig. 1. Time-weighted DET plot for the RT06s meeting room data. The solid line has a transition odds $R = 10$ in Eq. 1, the dashed line $R = 1$. Note that the DET axes have been extended to 90 % error probability.

re-segmentation process the silence regions found by the SAD are included to train a separate silence GMM model, and this model is included in the Viterbi segmentation.

One of the less satisfactory properties of the BIC method is that it has two tunable parameters λ_s and λ_c for the segmentation process and the stop-criterion, respectively. Merely by manual tuning we could improve devtest DER, not scoring overlapping speech, $e_{\mathrm{SD}}^{\mathrm{no}}$ from 34.2 % using $\lambda_s = 1.5$, $\lambda_c = 14$ (the parameters used for the RT05s evaluation) to 25.4 % by using $\lambda_s = 1.8$, $\lambda_c = 6$. The additional Viterbi re-segmentation further lowered $e_{\mathrm{SD}}^{\mathrm{no}}$ to 21.7 %.

The fact that the devtest results were so dependent on the two λ parameters did not give us very much confidence that the parameter setting would hold for RT06s, and as can be seen from Table 2 the evaluation results were rather poor, which is why we moved to better approached discussed below in section 3.2.

Table 2. Results of the BIC-based speaker diarization system for development test and evaluation test, both for evaluation condition without and with overlapping speech regions. For later reference, the processing speed (measured on an AMD Opteron 250 CPU) is indicated for the evaluation data.

Test set	Microphone	$e_{\mathrm{SD}}^{\mathrm{no}}$ (%)	e_{SD}^{o} (%)	Speed (× RT)
RT05s conference room	SDM	21.7	29.4	
RT06s conference room	SDM	32.4	44.8	0.79
RT06 lecture room	SDM	26.2	27.3	0.14
RT05s conference room	Array-beam	19.7	27.5	
RT06s conference room	Array-beam	35.6	46.3	
RT06 lecture room	Array-beam	47.6	48.7	

From the table an enormous degradation of performance can be observed going from devtest to the evaluation, specifically for the beam-formed microphone data.

Dealing with overlap. New in RT06s was that the primary evaluation measure included speech regions with overlapping speakers. This means that an ideal system should be capable of producing output where speakers overlap. Clearly, in the plain segmentation and clustering approach, there is at most one speaker in the output. We have experimented with a few algorithms to deal with this challenge. None of them resulted in lower e_{SD}^{o} on the devtest, but we want to briefly describe the idea.

Average overlap A trivial approach, with the sole purpose of potentially lowering the error rate, is based on the expectation that in the turn-taking process speakers may, on average, overlap. A trivial implementation is to post-process the output of a one-speaker diarization system such that boundaries between speakers are altered so that for a short time t_o speakers overlap.

This t_o would represent the average time of overlap between speakers, and may be tuned to give minimum e^o_{SD}. We found $t_o = 0$.

Two-speaker states A more interesting approach assumes that the features of the sum of two speech signals can be approximated by the sum of the features of the individual signals. Then starting with a segmentation of a one-speaker diarization system, we can build N state models for the individual clusters, and then add $\binom{N}{2}$ '2-speaker states,' which are trained using speech from pairs of clusters. The HMM topologycan then be extended with transitions from single speaker states to 2-speaker states that include that speaker, and vice versa. In decoding with this more complex model, the hope is that the GMMs of the 2-speaker state get activated when its speakers speak simultaneously. Although our implementation did produce overlapping speech, we observed that only in approximately 1/3 of the overlap time these were the correct speakers, while in the remaining cases the wrong speakers were produced. Hence, we could not lower e^o_{SD} this way.

3.2 Decoder based speaker diarization

In addition to the full-covariance single Gaussian BIC-based system, two decoder-based speaker diarization systems were submitted. Both contrastive speaker diarization systems consist of two modules. First the SAD module described in Sect. 2 filters out the non-speech parts of the audio signal. The second module, a Viterbi decoder, segments and clusters the speech, using diagonal covariance GMMs.

The decoder is an adapted version of the University of Twente 2006 decoder (UT06). The UT06 decoder is a Large Vocabulary Continuous Speech Recognition (LVCSR) decoder using Hidden Markov Models (HMM). Its state emission probabilities are calculated by Gaussian Mixture Models (GMM). The decoder uses the Perceptual Minimum Variance Distortionless Response (PMVDR) cepstral coefficients from the Sonic LVCSR toolkit [9]. For speaker clustering, the derivatives and energy features are not used and the decoder's lexical tree is replaced by a network of single state HMMs connected to each other by a single (non-emitting) start- and end-state. In the ideal situation, each state is trained on audio of exactly one unique speaker so that a final Viterbi alignment would result in an optimal segmentation and clustering of the speech.

In order to find an approximation to this ideal HMM topology, a straightforward algorithm is used. Initially the number of parallel states is chosen higher than the expected number of speakers. For conference meetings ten states are used. For lecture meetings the initial number of parallel states is five. The audio is randomly divided over the parallel states so that the HMM can be trained for the first time. In an iterative process, each state will split its Gaussian with the highest weight until the desired number of initial Gaussians is reached. This initial number of Gaussians is dependent on the total amount of data that is used to train the state. Experiments on the devtest data showed that the optimum number of training samples per Gaussian is 800. For the RT06s conference

meetings, this means that each state is initially trained with approximately 10–14 Gaussians. Making the number of Gaussians dependent on the duration of the meeting (the number of training samples) will ensure that the states are not under- or over-trained when the duration of the audio varies.

After the initial training iteration, a Viterbi alignment re-segments the audio for a new training iteration. This procedure is repeated until the overall Viterbi likelihood score does not improve any more. During this procedure, the states that initially were trained with data from various speakers will gradually change into states that are trained with data from a single dominating speaker. Because of small initial differences in the speaker distribution of each state, at the first iteration each state will be trained slightly better for one of the speakers compared to the others. By re-aligning the data, the states will attract data from these speakers. Each iteration the states will be trained on more data from the dominated speaker and on less from the other speakers.

Once each state is trained for the most part on a single speaker, the task remaining in order to obtain the ideal HMM topology is to reduce the number of states until there is exactly one state left for each speaker. The two systems use different methods to reduce the number of states. The following paragraphs will describe these methods.

HMM-BIC The HMM-BIC training approach is inspired by the ICSI-SRI Spring 2005 Diarization System described in [4]. The number of states is reduced by pairwise merging two states into one single state. The two states to merge are found by using the Bayesian Information Criterion (BIC).

For each combination of two states, a new state σ is trained containing the sum of the number of Gaussians in the two original states σ_a and σ_b. This *merged* state is trained using the training data of the original two states. From the Viterbi scores of these three models, the BIC score is calculated. Because the total number of Gaussians will not change if σ replaces σ_a and σ_b, the complexity of the system will not change. This makes it possible to calculate the BIC score for merging two models without the need for a penalty for model complexity which obsoletes the necessity for the parameter λ [10]. Let D_a be the data used to train model σ_a, and let D_b to train σ_b and D to train model σ, then

$$\mathrm{BIC}(\sigma_a, \sigma_b) = \log P(D|\sigma) - \log P(D_a|\sigma_a) - \log P(D_b|\sigma_b). \tag{5}$$

The BIC value is calculated for all possible pairs of states. If none of the BIC scores are higher than zero, merging is stopped. Otherwise the two states with the highest BIC score are merged. This effectively means that the two states are removed from the network and the merged state is placed into the network. The data of each speech segment (from the SAD) is re-aligned using this new network and another training iteration is performed. After that, the merging procedure is repeated.

Cut&Mix. Considering the merging of all combinations of two states, as described in the previous paragraph, takes a lot of computational effort. Another

disadvantage of the HMM-BIC method is that it is based on the assumption that the two states are trained with data from the same single speaker. Unfortunately when a state contains data from multiple speakers, this data is not spread over multiple states when the state is taken out of the system. The method of reducing states described in this section is developed in order to make the system faster and to make it more robust in cases a state is not trained solely using one speaker.

The basic idea of the Cut&Mix strategy is that all states are considered for removal, and that the state which improves the Viterbi likelihood most after removal is subsequently 'cut out,' and its Gaussians are redistributed over the remaining states.

Formally, at each iteration i the number of states in the HMM will be reduced by one, by sequentially removing state j. The overall Viterbi score L will be used to compare the original HMM topology $\mathcal{T}_i$ with the new topology $\mathcal{T}_{i+1}^j$. If the score $\max_j L(\mathcal{T}_{i+1}^j)$ is higher than the original score $L(\mathcal{T}_i)$, the new topology $\mathcal{T}_{i+1}^j$ is used in the next iteration as $\mathcal{T}_{i+1}$. If the maximum is lower than $L(\mathcal{T}_i)$, the new topology is discarded and the original HMM will be regarded as the optimum topology.

Once state j is to be removed from the topology, the number of Gaussians in the remaining states is increased until the total number of Gaussians in the original topology and in the new topology are equal.

In order to make a fair comparison between the Viterbi score of the original system $\mathcal{T}_i$ and the new HMM topology $\mathcal{T}_{i+1}$, the complexity of both systems should be the same. Therefore, the number of Gaussians of the new system should be increased until it is the same as the number of Gaussians in the original system. The Gaussians from the state that has been cut away are distributed over the GMMs in the same proportions as the speech data has been distributed by the Viterbi alignment. For each state in the HMM, the increase in assigned data from before cutting away state j and after cutting away state j is calculated. Also, the amount of data that was originally assigned to state j is divided by the number of Gaussians in this state. The result is the amount of increased data that a state needs in order to be assigned an extra Gaussian. If $N(x, i)$ is the amount of data (speech frames) used to train state x at $\mathcal{T}_i$ and $n(x)$ is the number of Gaussians in state x, then the number of extra Gaussians $\Delta n(k)$ for each state k at $\mathcal{T}_{i+1}$ is:

$$\Delta n(k) = \frac{N(k, i+1) - N(k, i)}{N(j, i)/n(j)} \tag{6}$$

Each GMM that is assigned new Gaussians will be re-trained. The new Gaussians are obtained by splitting the Gaussian with the highest weight. After all GMMs are re-trained, the new overall Viterbi score is calculated. This score will be compared to the original score. If this score is higher than the original, a new cutting iteration will be started.

RT-06 results. The speaker diarization error rates of the two decoder-based systems on the conference meeting audio are listed in Table 3. These systems only use the SDM microphones. On the devtest set, the data from RT05s, the Cut&Mix system outperforms the HMM-BIC system. This is not reflected in the results of the evaluation. As expected, the processing speed of the Cut&Mix system (real-time factor 2.25) is better than the speed of HMM-BIC (4.63). These factors are measured on an Intel Xeon 2.8 GHz processor.

Table 3. The speaker diarization results of the HMM-BIC and Cut&Mix decoder based systems.

		HMM-BIC		Cut&Mix	
Test set	Microphone	$e^{\text{no}}_{\text{SD}}$ (%)	e^{o}_{SD} (%)	$e^{\text{no}}_{\text{SD}}$ (%)	e^{o}_{SD} (%)
RT05s conference room	SDM	21.6	30.2	18.6	27.6
RT06s conference room	SDM	22.7	37.2	25.2	39.5
RT06 lecture room	SDM	30.8	32.4	30.1	31.6
Processing speed (×RT)		4.63		2.25	

Post evaluation analysis. During state reduction, the HMM-BIC method calculates all possible topologies with one less state. Once two states are merged, no further training of the HMM is needed and, using BIC, the best topology is chosen. The Cut&Mix method does not calculate all topologies before choosing the state to remove from the HMM. Although this method uses a good measure to pick the best state to remove, the remaining states need training after the choice has been made and only afterwards it can be determined if the new topology is indeed better than the original topology. When the original topology turns out to be better, the system will stop without considering cutting other states out of the HMM. Therefore it is possible that the system stops reducing states too soon and that the clustering result contains too many speakers.

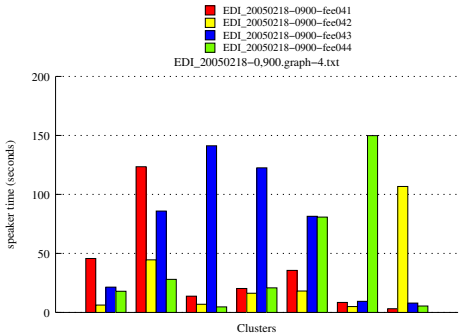

Fig. 2. Final speaker distribution of meeting EDI-20050218.

Figure 2 is a graphical representation of a situation where this is the case. It shows seven clusters (states) that contain the amount of speaker time of each speaker in the meeting that is assigned to the cluster. During the final attempt to reduce states, the first cluster was chosen to be cut away. This resulted in a worse performing topology and therefore the seven clusters were maintained. Cutting away the third or fourth state probably would have resulted in a better topology. The speaker diarization error-rate without overlapping speech of the HMM-BIC system on this meeting was 29.5%. On the Cut&Mix system this was 44.9%. We believe that this shortcoming in the stop criterion of the Cut&Mix system is the main reason why it did not perform as expected.

For both systems, after the initial HMM training iteration, each state should contain a dominating speaker. If this is not the case, reducing the states will not always result in a clean final clustering. States trained on multiple speakers may remain in the system (like for example the fifth cluster in figure 2) and speakers that were not dominant in any state during the initial training iteration, may never earn their own state. In Fig. 3 the speaker distribution in one of the conference meetings after the initial training iteration is drawn (graph at the left). Speaker 'vhqqmy' (the rightmost bar in each cluster) is not dominating in any of the clusters. The graph at the right in figure 3 shows the speaker distribution after the final iteration. At this point speaker 'vhqqmy' is still not assigned its own state. Also, the first two states contain data from multiple speakers. It is possible that states with multiple speakers contain a considerable amount of overlapping speech. This would explain why data from these states is not assigned to states with dominating speakers (e.g., the fourth state in Fig. 3, right). Apart from this problem, a better method for determining the initial speaker distribution might improve system performance considerably.

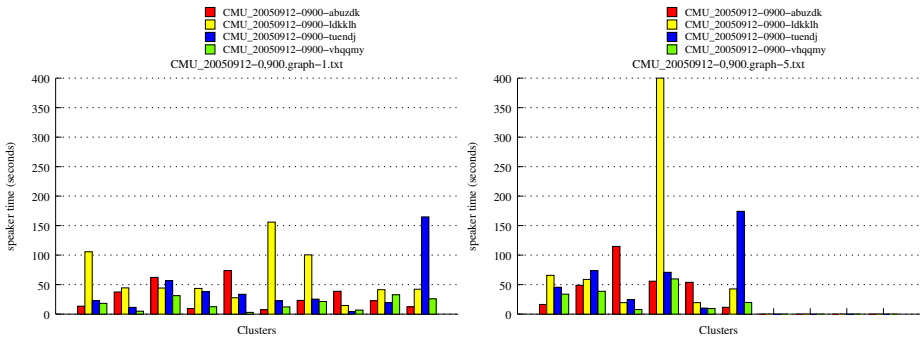

Fig. 3. Speaker distribution of meeting CMU-20050912. Left: distribution after the first training iteration. Right: distribution after the final training iteration.

4 Discussion and Conclusions

The performance of our SAD system, when expressed in time-weighted error terms e_{SAD}, seems consistently performing very well for the meeting room evaluation data. For both RT05s and RT06s this relatively straightforward system showed low error rates. As a pre-processing step to a SPKR system it functions adequately. Our AMI colleagues [3] have tried to use the SAD output for the speech-to-text MDM task, but this lead to practical problems such as a wide range of segment sizes that were difficult to deal with. Note that all evaluation measures discussed here for SAD and SPKR tasks do not evaluate the frequency at which (erroneous) segment boundaries occur. Very many spurious error segments, if small in duration, will not add much to the error rates, while such a result may be completely useless for interpretation by person or machine.

The evaluation prior $p(\mathrm{speech})$ appears to be very important in choosing the operating point for the SAD decoder. In the devtest set this prior was 0.955, for RT06s it turned out to be 0.947, very close. In speech recognition, another field where decoding plays an essential role, it is normal to include prior information in the system (e.g., the language model), while for speaker detection it is customary to take the prior out of the detector. When SAD is seen as a detection problem, it might be more diagnostic to work with detection-based evaluation measures, such as p_{FA}, p_{miss} and analyze the detector in terms of post-evaluation measures such as the Equal Error Rate. We have shown that the DET trial-based framework can be extended to a time-weighted segmentation-based framework. Our first experiments show that the obtained DET curves look reasonable (a 'straighter' DET curve indicates that the underlying score distributions are 'more Gaussian' [11]), but changing detection thresholds after scores have been produced is not the same as choosing different segmentation parameters, such as our R. In a way, this approach resembles the evaluation of word spotting systems, where also segmentation (the spotting of words) is carried out including the production of scores (acoustic likelihoods).

The SAD performance for lecture room meetings, however, was far below average. We have suggested that this may be the result of acoustic mismatch of the models. This may have been the cause of our poor SPKR results for the lecture room meetings, but we are not convinced that the lecture room domain is an interesting problem for speaker diarization, see for instance the 'one speaker takes all' approach of ICSI for RT05s [4].

In our development of different SPKR systems, it has become clear that the popular BIC segmentation method, originally developed for Broadcast News domains, shows severe problems in terms of the tuning of the complexity penalty parameter λ. The two approaches based on the ICSI system of RT05s, the standard HMM-BIC and the derived Cut&Mix systems use Gaussian mixtures to keep the complexity of the system constant so that the systems do not need tunable parameters. These systems have proven to be more robust against new evaluation data. One might argue that the single Gaussian BIC segmentation method still has advantages, such as computational efficiency. The system runs easily under $1 \times$ RT, the figures reported in Table 2 are high estimates because

we ran our system for the entire meeting, rather than only the segments indicated in the evaluation index files. However, we believe that the sensitivity of the choice of the λ to the application domain needs proper attention.

Concentrating now on the two decoder-based speaker diarization system, the RT06s evaluation has shown that our HMM based approach is competitive to other systems. The Cut&Mix system is not performing as well as the HMM-BIC system, probably because of its poor stop criterion. We are planning to test other stop criteria for the Cut&Mix system in order to improve its performance without increasing the processor load.

Analysis of the SPKR evaluation results show that the initial speaker distribution affects the final distribution. Speakers that are not dominating any state in the initial training run will not likely be assigned a private state during the following iterations. Some states that contain about the same amount of data of multiple speakers will not converge to a single speaker. In future we will investigate if this problem is caused by overlapping speech. Also we will investigate new methods for distributing data for the initial training iteration.

Acknowledgements

This work was partly supported by the European Union 6th FWP IST Integrated Project AMI (Augmented Multi-party Interaction, FP6-506811, publication). It is further partly supported by the project MultimediaN (`http://www.multimedian.nl`). MultimediaN is sponsored by the Dutch government under contract BSIK 03031.

References

1. Fiscus, J.G., Radde, N., Garofolo, J.S., Le, A., Ajot, J., Laprun, C.: The rich transcription 2006 spring meeting recognition evaluation. Lecture Notes in Computer Science (2007) Submitted for publication.
2. van Leeuwen, D.A.: The TNO speaker diarization system for NIST rich transcription evaluation 2005 for meeting data. Lecture Notes in Computer Science (2006) 400–449
3. Hain, T., Burget, L., Dines, J., Garau, G., Karafiat, M., Lincoln, M., Vepa, J., Wan, V.: The AMI meeting transcription system: Progress and performance. Lecture Notes in Computer Science (2007) Submitted for publication.
4. Anguera, X., Wooters, C., Peskin, B., Aguiló, M.: Robust speaker segmentation for meetings: The ICSI-SRI spring 2005 diarization system. Lecture Notes in Computer Science (2006) 402–414
5. Hermansky, H., Morgan, N.: Rasta processing of speech. IEEE Transactions on Speech and Audio Processing, special issue on Robust Speech Recognition **2** (1994) 578–589
6. Macho, D., Temko, A., Nadeu, C.: Robust speech activity detection in interactive smart-room environment. Lecture Notes in Computer Science (2007) Submitted for publication.

7. Martin, A., Doddington, G., Kamm, T., Ordowski, M., Przybocki, M.: The DET curve in assessment of detection task performance. In: Proc. Eurospeech 1997, Rhodes, Greece (1997) 1895–1898
8. van Leeuwen, D.A., Bouten, J.S.: Results of the 2003 NFI-TNO forensic speaker recognition evaluation. In: Proc. Odyssey 2004 Speaker and Language recognition workshop, ISCA (2004) 75–82
9. Pellom, B., Hacioglu, K.: Recent Improvements in the CU Sonic ASR system for Noisy Speech: The SPINE Task. In: Proc. ICASSP. (2003)
10. Ajmera, J., McCowan, I., Bourlard, H.: Robust speaker change detection. IEEE Signal Processing Lettres **11** (2004) 649–651
11. Navrátil, J., Ramsawamy, G.N.: The awe and mistery of t-norm. In: Proc. Eurospeech. (2003) 2009–2012