INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

DNN-based speaker clustering for speaker diarisation

Rosanna Milner, Thomas Hain

Speech and Hearing Research Group, University of Sheffield, UK
`rmmilner2, t.hain@sheffield.ac.uk`

Abstract

Speaker diarisation, the task of answering "who spoke when?", is often considered to consist of three independent stages: speech activity detection, speaker segmentation and speaker clustering. These represent the separation of speech and non-speech, the splitting into speaker homogeneous speech segments, followed by grouping together those which belong to the same speaker. This paper is concerned with speaker clustering, which is typically performed by bottom-up clustering using the Bayesian information criterion (BIC). We present a novel semi-supervised method of speaker clustering based on a deep neural network (DNN) model. A speaker separation DNN trained on independent data is used to iteratively relabel the test data set. This is achieved by reconfiguration of the output layer, combined with fine tuning in each iteration. A stopping criterion involving posteriors as confidence scores is investigated. Results are shown on a meeting task (RT07) for single distant microphones and compared with standard diarisation approaches. The new method achieves a diarisation error rate (DER) of 14.8%, compared to a baseline of 19.9%.

Index Terms: speaker diarisation, speaker separation, deep neural network

1. Introduction

Speaker diarisation is the task of answering "who spoke when?" in an audio recording [1, 2]. Three stages are considered: speech activity detection, where speech-only segments of time are found; speaker segmentation, or speaker change detection, where these segments are split into speaker homogeneous segments; and finally speaker clustering, in which the segments are grouped into same-speaker clusters. Diarisation has been well studied over the years, and several toolkits are available for this task, however most are designed to perform well for a specific type of data [4, 5, 6]. The most common method of speaker clustering is agglomerative hierarchical clustering (AHC), also known as bottom-up clustering, using the Bayesian information criterion (BIC) as the decision metric and stopping criterion. This can occur in iterations of Viterbi realignment using speaker models based on the previous clustering iteration [3]. This paper presents a novel method of speaker clustering using deep neural networks. Traditionally being an unsupervised task, speaker clustering with discriminate classifiers, such as DNNs has not been successful. This is due to the difficulty discriminate classifiers have with mislabelled data.

In previous work integrating neural networks into a speaker diarisation system, a speaker segmentation stage using auto-associative neural networks (AANN) was proposed [7]. A windowing method is used where an AANN model is trained for the left half of the window and tested on the right to give a confidence score on how likely each part belongs to the same speaker. More recently, artificial neural networks (ANN) have been trained to learn a feature transform [8, 9]. DNN senone posteriors have also been combined with i-vector extraction [10]. The method we present involves training a DNN which learns how to separate speakers. DNNs which can classify speakers have been implemented in the speaker recognition field [11] which involves projecting acoustic features into a lower-dimensional feature set. Our own previous work has also implemented speaker separation DNNs [12].

The method proposed in this paper was partially introduced in our diarisation system [12] and ASR system [13] for the MGB challenge 2015 [14]. DNNs are iteratively adapted, where a previously trained speaker separation DNN is retrained using an initial clustering output. The DNN resegments and reclusters the data. This gave an absolute DER improvement of 2%. However, this configuration is limited to situations where an initial clustering is available. This paper presents the extension of this adaptation to speaker clustering, having removed the need of a prior clustering. For implementing this novel speaker clustering method, two prerequisites are necessary: a DNN trained to classify speakers; and a speech/non-speech segmentation of the audio file. This speech/non-speech segmentation can contain speaker-homogeneous segments or be speech-only. A DNN for each audio file is iteratively built and adapted given the input segments. The segments are relabelled every iteration, with an option to allow speaker boundaries to occur within the speech segments. This is speaker segmentation. Methods of filtering the data used for adaptation allows to improve purity. Benefits of an automatic stopping criterion over a fixed number of iterations is investigated. The method is evaluated on an established test set, meeting data from the NIST Rich Transcription evaluations in 2007 (RT'07) [16].

The proposed method is presented in Section 2, which consists of training the speaker separation DNN, reconfiguring the final layer and finally the stopping criterion. Methods of filtering the input data the segmentation used for adaptation are detailed in Section 3. The experimental setup is covered in Section 4 and finally, the results are shown in Section 5.

2. Speaker clustering using DNNs

The DNN-based speaker clustering method is semi-supervised as it requires a speaker separation DNN (ssDNN) to be trained beforehand. Also, speech segments are required as input into the clusterer. The algorithm in depicted in Figure 1. The iterative process begins by building new DNNs based on the ssDNN, one for each audio file. The final layer of the ssDNN is removed and replaced and adaptation is carried out using the input speech segments. Decoding is carried out on these same segments which can allow speaker segmentation and the data is relabelled. If the stopping criterion is met, the relabelled segments are the final output, otherwise the process is repeated using the output segments as the new segments for updating the network in the next iteration.

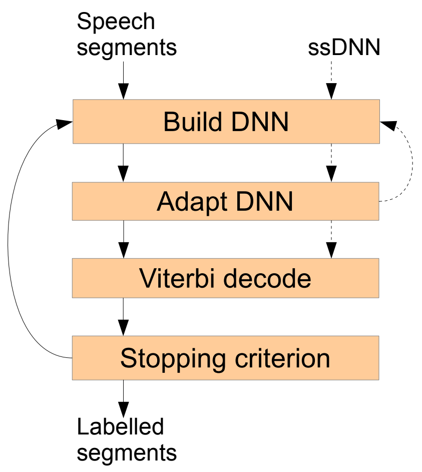

Figure 1: The algorithm is depicted where the solid line represents the input speech segments and the dashed line represents the ssDNN.

2.1. Speaker separation DNN (ssDNN)

For the semi-supervised part of the method, a DNN is trained which learns how to separate, or classify, speakers. This is carried out by training a network which has several hidden layers, including a bottleneck layer. This bottleneck layer aims to capture the key information necessary to classify what has been learnt, in this case the target classes are speakers. It is assumed that the network is working in a similar way as to how bottleneck features are produced, such as i-vectors [15]. This assumption leads to the next stage in the method, where the final layer is replaced.

2.2. Reconfiguring the final layer

For each audio file, new DNNs are derived from the ssDNN. The final layer is removed and replaced by an initial untrained final layer. The use of the ssDNN as a starting point is to use its knowledge of how to separate different speakers, ideally captured at the bottleneck.

The built networks are then adapted using the provided speech segments. These could be speaker homogeneous or speech only segments. The initial iteration begins by adapting the built DNN with these segments, where each segment has a distinct label, i.e. a label representing the segment number. These are the target classes for the DNNs. This is equivalent to the standard approach of bottom-up clustering where each segment begins as a separate cluster. This adaptation of the built networks can occur in any number of adaptation iterations, as opposed to iterations of the algorithm. However, it was found that a single iteration of the adaptation stage was sufficient.

Viterbi decoding occurs on the same speech segments used for adapting the DNNs. Decoding can occur frame-by-frame which allows the input speech segments to split to obtain speaker boundaries. This produces a relabelled and resegmented output of the input speech segments. The method implemented in this way allows for speaker segmentation at the same time as speaker clustering. If wishing to keep the same segment boundaries and prevent speaker segmentation from occurring, the log probabilities for each segment can be considered. The class with the highest average log probability per frame is chosen for that segment.

If the stopping criterion is met, the relabelled segments output from the decoding stage are the final output. Otherwise, the iterative process begins again. The target classes for the next iteration, the number of nodes in the final layer, are the classes found in the decoding output. The data for adaptation is the labelled output segments. If a target class has found no data in the decoding, the class is removed from the adaptation. This means classes can be lost, but never gained.

2.3. Stopping criterion

As in all clustering algorithms, it is key to know when to cease iterations of clustering. In speaker diarisation, the aim is to stop clustering when the correct number of speakers is reached, but this is unknown. Here each audio file is treated independently. A simple method for stopping would be to stop training after a set number of iterations, this being the same for each audio file.

A more advanced stopping criterion varies for each audio file, such as the ΔBIC in standard methods. Several options exist for choosing a stopping criterion: an average probability per frame; the minimum probability; or the maximum probability across all segments. The percentage of change between the current iteration and previous iteration can be calculated. Once the change is less than a decided threshold, the iterations cease. The lower the change implies the system is heading towards convergence. Best results were obtained using the percentage of change of the average probability per frame between iterations.

3. Data Filtering

Discriminant classifiers behave badly when presented with impure data. The speech segments used for adaptation are vital to be as pure as possible, in terms of speaker error. Small errors here would lead to errors being learnt by the DNNs and potentially increasing every iteration. Selecting pure segments for adaptation would give the system a better starting point. This can occur in two parts of the system: the input speech segments; and decoding the segments.

3.1. Improving purity

There are various ways to refine and improve upon the input segments used for adaptation. The input speech segments in the first iteration are from the provided speech/non-speech segmentation, where the speech-only segments are used. All further iterations derive the adaptation data from the decoded output in the previous iteration.

Short segments can cause problems as they contain little information to base an entire speaker on each in the first iteration. One option is to remove them completely from the adaptation stage. This allows longer segments, which contain more information, to be used. The decoding will still occur using all the segments but these will be filtered by time for each adaptation stage.

Further to filtering by time, DNNs do work better when presented with more data. Knowing initially the short segments could disrupt the adaptation, with each iteration the amount of filtered short segments could be reduced. This assumes each adaptation iteration is getting closer to stability and convergence.

If the provided speech-only segments are allowed to split (creating speaker boundaries) during the decoding, then these provided segments which have been given more than one class can be removed from the next adaptation iteration. This could help to reduce impure segments being given to the DNN for it to adapt to, avoiding adapting to incorrect data.

3.2. Decoding

Decoding is carried out on all the provided speech-only segments. There are several ways to improve and tune the decoding stage. First, a fixed state duration using hidden Markov models (HMMs) can be enforced. It must be noted that a state duration longer than a segments length will cause this segment not to be decoded, thus removed from the output. This can cause missed speech in the final scoring. To avoid this missed speech, any segment less than the state duration can be decoded frame by frame. The prior probabilities for each target class can also be varied, opposed to keeping every class at an equal probability of occurring. This allows classes represented by more data to have a higher probability and thus be more likely to be selected for a segment. Lastly, the grammar scale factor can be varied. Changing this helps to balance acoustic and language model scores.

4. Experimental Setup

The configuration for the ssDNN training, DNN adaptation and experiments are described below. The data used as test along with the evaluation metrics and baseline systems are also described.

4.1. Data

Experiments are carried out on NIST RT'07 meeting data [16]. This is a standard dataset for speaker diarisation produced by NIST for evaluation purposes. Single distant microphone (SDM) audio is used and the dataset consists of 8 meetings with a total of 35 speakers. Manual reference segmentation has been produced for RT'07 to the accuracy of 0.1 second. This means the results displayed on this paper are not comparable with previous published RT'07 results. The updated manual reference is avaliable from our website[1]. This has allowed for completely speaker pure segments to be used in our DNN adaptation, and the overlapping portions have been removed. There are 10847 non-overlapping segments.

4.2. DNN training

DNNs were trained using filterbanks of 23 dimensions with a context window of 16 frames on both sides. Log Mel-filterbanks are used as opposed to Mel frequency cepstral coefficients (MFCCs) as they are found to yield better performance with DNNs [18]. The ssDNN is trained on meeting data in the AMI corpus [19]. Individual headset microphone audio (IHM) is used. The input layer has 368 nodes, four hidden layers have 1745, and a bottleneck layer is used with 13 nodes. The final output layer has 183 nodes, representing 183 speakers. Different ssDNNs were tested where the number of hidden layers, size of the hidden layers and the size of the bottleneck was varied. The configuration described produced the lowest DERs.

4.3. Measuring diarisation performance

Diarisation error rate (DER) is the standard metric for speaker diarisation. It is the sum of three frame error values: miss (MS), false alarm (FA) and speaker error (SE) [1, 2, 20]. Missed speech refers to reference speech detected as silence, false alarm is reference silence detected as speech, and speaker error measures the percentage of scored time in which a speaker label is assigned to the wrong speaker. The definition of DER was established by NIST [16], which also includes a "collar", typically 0.25 seconds. The reason for the collar is to allow for human errors in the reference, any error values inside the collar time are not counted. However, as the manual reference segments are used, the collar is set to 0.0 seconds.

[1]mini.dcs.shef.ac.uk/resources/dia-improvedrt07reference/

Baseline	DER	#spkrs	#segs
speaker-cluster	19.9	29	10689
SHoUT	26.6	41	11522

Table 1: Two baseline methods showing the DER, number of speakers (#spkrs) and number of segments (#segs).

4.4. Baseline experiments

Two baselines are implemented. The first is our own speaker clustering tool, referred to as speaker-cluster, which implements a standard method in the field of speaker diarisation: bottom-up clustering using BIC as the decision metric and as the stopping criterion. This does a single pass of speaker clustering and thus does not resegment the data. The second is a public domian toolkit, SHoUT [6]. It is designed for diarisation of meetings and uses BIC segmentation and BIC stopping criterion in an unsupervised model training regime. The manual reference segmentation is given which runs a rough clustering for initial speaker models and then resegments the data using these models in several passes. This changes the speech/non-speech segmentation.

5. Results

Results have been divided into three types of experiments: baselines, fixed number of training iterations, and invoking a stopping criterion. The latter two result in no missed or false alram speech due to using the reference segmentation.

5.1. Baselines

The results for the two baselines are shown in Table 1. The speaker-cluster tool achieves a DER of 19.9%. As it does not resegment the input, it produces 0.0% FA but it does produce 0.1% MS due to ignoring segments of little length (less than 3 frames). However, SHoUT does resegment in terms of speech/non-speech so the overall DER of 26.6% is higher than the speaker cluster tool due to causing MS and FA. It is not a comparable number because of this issue, so the baseline of 19.9% will be used.

5.2. Fixed iterations

Results for the experiments which stop after a fixed number of training iterations are shown in Table 2, and the scores are taken after 5 iterations. Figure 2 displays results from 10 iterations of the final experiment. Two audio recording results are displayed. They show the progression of the DER, the number of speakers and probability scores. The fifth iteration was decided upon as the number of speakers settles across the files at iteration 5.

A first experiment was performed where the provided segments where forced not to split, speaker segmentation was prevented. Despite the segmentation being the reference, it performs poorly. This is compared to allowing the input speech segments to split. This improves the DER down to 29.8% and finds more speakers. It does however more than double the number of segments produced. Next, at each iteration, 25% of the time in the shortest segments is removed, again reducing the DER. Both the number of speakers and number of segments increase. Implementing a state duration of 30 states, reduces the number of segments, speakers and DER. In the penultimate experiment, reducing the amount of time filtered and removing the split segments from the training at each iteration produces further gains, although the number of speakers drops below the reference number. The last experiment changed the grammar scale factor from 1 to 6. This causes a decrease in DER to 14.9% and has found the correct number of speakers overall.

System	DER	#spkrs	#segs	#iters
51.368.1745.13:nosplits	47.4	29	10847	40
51.368.1745.13:splits	29.8	38	27545	40
+timefilter=0.25	27.8	54	35230	40
+states=30	21.8	49	15675	40
+reducefilt+rmsplits	19.3	32	15270	40
+gsf=6	14.9	35	13090	40

Table 2: Results with a fixed number of 5 training iterations showing the DER, number of speakers (#spkrs), number of segments (#segs) and the number of iterations across all files (#iters).

System	DER	#spkrs	#segs	#iters
51.368.1745.13:nosplits	49.9	29	10847	48
51.368.1745.13:splits	29.8	38	27510	40
+timefilter=0.25	27.7	54	34765	46
+states=30	21.1	49	15302	56
+reducefilt+rmsplits	18.3	31	14943	53
+gsf=6	14.8	35	13024	50

Table 3: Results using the automatic stopping criterion showing the DER, number of speakers (#spkrs), number of segments (#segs) and the number of iterations across all files (#iters).

The number of split speech segments has reduced.

Overall, the DER has successfully decreased at each stage and the last three experiments have achieved results lower than the baselines. After filtering the training segments by time, each further experiment reduces the number of segments until the final experiment which over segments by 10%, a drop from 69%. Lastly, if the final experiment is continued for 50 iterations, the DER reaches 13.0%, stopping at 5 iterations is a loss of 1.9%.

5.3. Stopping criterion

Table 3 shows the results for the same experiments in Table 2, however the automatic stopping criterion is put in place. A threshold of 1% is applied, meaning if the percentage of change between two iterations is less than this threshold, the current iteration is chosen as the final iteration. This allows a similar number of iterations to occur. Similar progress through experiments is seen here, however in most stages the DER is lower than (or equal to) the equivalent experiment with a fixed number of iterations. The penultimate experiment gives the largest gain, from 19.3% to 18.3%, a 1% gain. This shows implementing a stopping criterion produces better or same results than a fixed number of iterations. Unfortunately this is not true for the first experiment as the DER increases, however the DER is already large and problematic at nearly 50%.

As previously mentioned, the final experiment was continued for 50 iterations. For each audio file, the iteration with the lowest DER was manually selected and these were scored. The overall DER reaches 12.9%, this score is the goal of the automatic stopping criterion with a large number of iterations to choose from. Error of 1.9% is gained in the suggested stopping criterion which is produced with a lower number of iterations. In terms of computation, 1.9% DER is gained from having the ability to use a large number of iterations and less computation increases the DER.

With the stopping criterion in place, Recording7 stops at iteration 8 and Recording8 stops at iteration 9 in Figure 2, when the percentage of change in scores is less than the 1% threshold. The DER is not stable, but the change is small after the fifth iteration and Recording8 shows this may not always be a reduction in DER over time. However, the stopping criterion helps to reduce the DER or keep it the same as a fixed number of iterations.

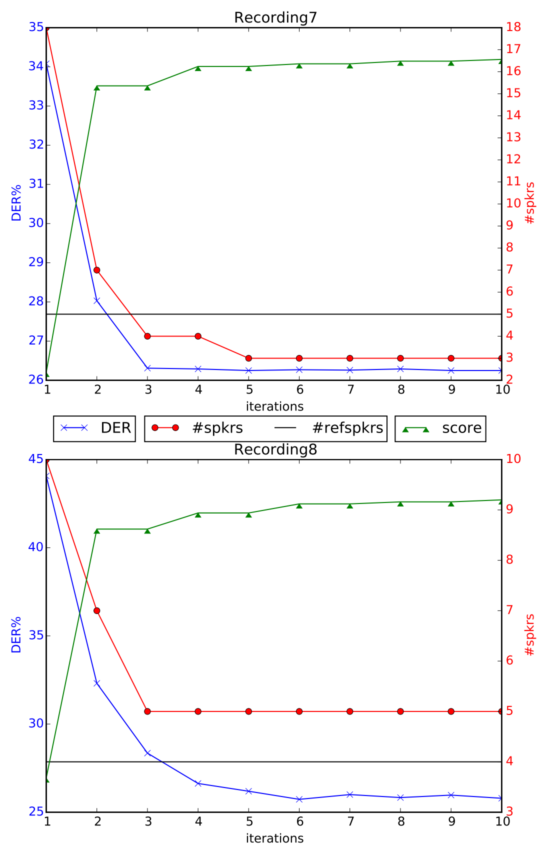

Figure 2: Results for two audio files: the DER, number of speakers (#spkrs), number of reference speakers (#refspkrs) and average probabilities (score) for each iteration.

6. Conclusions

A novel semi-supervised speaker clustering technique for speaker diarisation is proposed which uses DNNs. It requires a pretrained speaker separation DNN which is used as a starting point for building new DNNs for each audio file. The newly built DNNs are then iteratively adapted using speech segments. An average probability per frame stopping criterion, involving change across iterations, allows for the training iterations to automatically stop. It is shown to reduce or keep the same DER score in a similar number of iterations. Experiments involved filtering the adaptation segments and improving the decoding stage, both in order to produce purer segments to readapt the DNN to. The experiments show a fixed number of training iterations reaches 14.8% as well as experiments invoking stopping criterion. This is lower than the baseline of 19.9%. A large number of iterations, 50, can reduce the DER by 1.9% in the final experiment, if computation is not an issue.

7. Acknowledgements

We would like to thank Jana Eggink and the BBC for supporting this work. This work was also supported by the EPSRC Programme Grant EP/I031022/1 Natural Speech Technology. Results can be found here: https://dx.doi.org/10.6084/m9.figshare.3457523

8. References

[1] S. E. Tranter, “Who Really Spoke When? Finding Speaker Turns and Identities in Broadcast News Audio,” *ICASSP*, vol. 1, pp. 1013–1016, 2006.

[2] X. M. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, “Speaker Diarization: A Review of Recent Research,” *IEEE Trans. Audio Speech and Language Processing*, vol. 20, no. 2, pp. 356–370, Feb. 2012.

[3] X. Anguera, C. Wooters, B. Peskin, and M. Aguil, “Robust speaker segmentation for meetings: The ICSI-SRI spring 2005 diarization system,” in *In Proc. of NIST MLMI Meeting Recognition Workshop*, 2005, pp. 26–38.

[4] D. Vijayasenan and F. Valente, “DiarTk: An Open Source Toolkit for Research in Multistream Speaker Diarization and its Application to Meetings Recordings.” *INTERSPEECH*, pp. 5–8, 2012.

[5] M. Rouvier, P. Gay, E. Khoury, T. Merlin, S. Meignier, and L. Mans, “A Free State-of-the-art Toolbox for Broadcast News Diarization,” *INTERSPEECH*, 2013.

[6] M. A. H. Huijbregts, “Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled,” Ph.D. dissertation, 2008.

[7] S. Jothilakshmi, V. Ramalingam, and S. Palanivel, “Speaker diarization using autoassociative neural networks,” *Eng. Appl. of AI*, vol. 22, no. 4-5, pp. 667–675, 2009.

[8] S. H. Yella, A. Stolcke, and M. Slaney, “Artificial neural network features for speaker diarization,” in *2014 IEEE Spoken Language Technology Workshop, SLT 2014, South Lake Tahoe, NV, USA, December 7-10, 2014*, 2014, pp. 402–406.

[9] S. H. Yella and A. Stolcke, “A comparison of neural network feature transforms for speaker diarization,” in *Proc. Interspeech*. Dresden: ISCA - International Speech Communication Association, September 2015.

[10] G. Sell, D. Garcia-Romero, and A. McCree, “Speaker diarization with i-vectors from DNN senone posteriors,” in *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, 2015, pp. 3096–3099.

[11] Y. Konig, L. Heck, M. Weintraub, K. Sonmez, and R. E. E, “Nonlinear discriminant feature extraction for robust text-independent speaker recognition,” in *In Proc. RLA2CESCA Speaker Recognition and its Commercial and Forensic Applications*, 1998, pp. 72–75.

[12] R. Milner, O. Saz, S. Deena, M. Doulaty, R. Ng, and T. Hain, “The 2015 Sheffield System for Longitudinal Diarisation of Broadcast Media,” in *Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Scottsdale, AZ, 2015.

[13] O. Saz, M. Doulaty, S. Deena, R. Milner, R. Ng, M. Hasan, Y. Liu, and T. Hain, “The 2015 Sheffield System for Transcription of Multi–Genre Broadcast Media,” in *Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Scottsdale, AZ, 2015.

[14] P. Bell, M. Gales, T. Hain, J. Kilgour, P. Lanchantin, A. Liu, A. McParland, S. Renals, O. Saz, M. Wester, and P. Woodland, “The mgb challenge: Evaluating multi-genre broadcast media recognition,” in *Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Scottsdale, AZ, 2015.

[15] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE Trans. Audio, Speech & Language Processing*, vol. 19, no. 4, pp. 788–798, 2011. [Online]. Available: http://dx.doi.org/10.1109/TASL.2010.2064307

[16] “The NIST Rich Transcription Evaluations,” http://www.itl.nist.gov/iad/mig/tests/rt/, accessed: 16-07-2014.

[17] K. Vesel, “Parallel training of neural networks for speech recognition,” in *Proceedings of the 16th Conference STUDENT EEICT 2010*, ser. Volume 3. Brno University of Technology, 2010, pp. 74–76.

[18] H. Hermansky and S. Sharma, “Traps – Classifiers Of Temporal Patterns,” in *ICSLP*, 1998, pp. 1003–1006.

[19] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner, “The AMI Meeting Corpus: A Pre-announcement,” vol. 3869, pp. 28–39, 2006.

[20] “Diarisation error rate scoring code, NIST,” http://http://www.itl.nist.gov/iad/mig/tests/rt/2006-spring/code/md-eval-v21.pl, accessed: 16-07-2014.