AUTOMATIC SPEAKER CLUSTERING FROM MULTI-SPEAKER UTTERANCES*

Jack McLaughlin, Douglas Reynolds, Elliot Singer and Gerald C. O'Leary

MIT Lincoln Laboratory

Lexington, MA 02420-9185

jackm | dar | es | gco @sst.ll.mit.edu

ABSTRACT

Blind clustering of multi-person utterances by speaker is complicated by the fact that each utterance has at least two talkers. In the case of a two-person conversation, one can simply split each conversation into its respective speaker halves, but this introduces error which ultimately hurts clustering. We propose a clustering algorithm which is capable of associating each conversation with two clusters (and therefore two-speakers) obviating the need for splitting. Results are given for two speaker conversations culled from the Switchboard corpus, and comparisons are made to results obtained on single-speaker utterances. We conclude that although the approach is promising, our technique for computing inter-conversation similarities prior to clustering needs improvement.

1. INTRODUCTION

In [1], we described approaches to blind clustering of speech utterances based on speaker and language characteristics. For speaker attribute clustering, it was assumed that a single speaker was speaking in each utterance and so exhaustive, exclusive clustering, where each message was placed in one and only one cluster, was allowable. However, in the more general case, recorded messages, such as from meetings, will contain speech from more than one speaker so that non-exclusive clustering, where a message can be placed in more than one cluster, is required.

One way to treat this problem is to attempt to segment each conversation into single-speaker utterances as done in [2]. Any one of a number of clustering algorithms can then be applied. This split-then-cluster approach was employed here on two-speaker telephone speech from the Switchboard corpus, and the result (see Table 1) serves as our baseline. The conversation splitting was achieved by agglomerating one-second segments into two clusters based upon scores for each segment against models for every other segment. Follow-on clustering was done using the d* algorithm [1].

Because the splitting operation is imperfect, subsequent clustering results in more errors than when conversations are split using truth markings when such markings are available. Experiments performed here at Lincoln Laboratory verify this (see again Table 1).

Here, we propose a clustering technique which is capable of placing each conversation into exactly two clusters, obviating the need for speaker splitting and its inherent error. This clustering method, adapted from the widely known agglomerative clustering algorithm, operates on an inter-conversation similarity matrix computed using log likelihood scores output from a Gaussian mixture model (GMM) speaker identification system.

In the following section, we describe this similarity computation prior to describing our two-speaker agglomerative clustering algorithm in section three. Our agglomerative algorithm produces a tree structure, as does the standard agglomerative, and this tree must be cut to produce clusters. To assess the goodness of our cut, we use a modification of the BBN metric described in [3]. This modification and our results comprise the final sections.

2. SIMILARITY MATRIX COMPUTATION

For two-speaker conversation clustering, we would like our similarity metric to have the property that conversations with a common speaker will have high similarity and conversations with no common speaker will have low similarity. Our similarity metric is based upon our GMM speaker identification system [4]. In this case, we wish to compute a similarity between every pair of conversations. We proceed by forming a 2048-mixture GMM (using cepstral and delta-cepstral features) for each conversation. This GMM is adapted from a universal background model (UBM) as described in [5]. Both speakers in the conversation are modeled using this single GMM.

*THIS WORK WAS SPONSORED BY THE DEPARTMENT OF THE AIR FORCE. INTERPRETATIONS, CONCLUSIONS, AND RECOMMENDATIONS ARE THOSE OF THE AUTHORS AND ARE NOT NECESSARILY ENDORSED BY THE UNITED STATES AIR FORCE.

The similarity between two conversations A and B is obtained by computing the likelihood of each one-second segment of conversation B scored against the model for conversation A *and* the UBM, and then forming a likelihood ratio. The final similarity is the sum of all segments above a certain threshold. This threshold, which we set to zero, allows us to reject all segments of B which have no speaker in common with the two speakers in A.

Note that in this procedure, there is no normalization for the duration of each speaker's speech in a conversation. Though such normalization may be desirable, it has a downside. The use of the threshold to reject segments that have no common speaker is prone to error and even though several segments might contribute to the final similarity, it may still be that in fact, there is no common speaker. If we were to normalize by, say, the number of contributing segments, we would lose this information. In failing to normalize, we indirectly preserve information about the number of segments that contributed to the sum.

3. AGGLOMERATIVE CLUSTERING

Tree Formation

Agglomerative clustering (see, for example, [6]) is a proven technique for blind clustering [1][3]. With the standard algorithm, all items to be grouped are initially considered as being in their own clusters. (All items are singletons.) A distance (or similarity) exists between each item and every other. With each iteration, the two items which are most similar are merged to form ever larger non-singleton clusters. Upon the merging of two clusters, the need arises to compute a distance (or similarity) between the new cluster and every other item. There are a number of reasonable options, but we choose to define the similarity between two clusters as equal to the similarity between the least similar conversations in those clusters. Agglomeration of clusters continues until there is only a single large cluster remaining. We wish to adapt this approach for the multi-speaker case.

Initialization

In clustering of two-speaker conversations, we do not wish to cluster the conversations themselves so much as we wish to cluster the speakers in those conversations. Since we assume that each conversation has two speakers, the initial number of clusters is equal to twice the number of conversations with each conversation belonging to two unique clusters. However, similarities exist only between every pair of conversations, not every pair of speakers. This situation is illustrated in Figure 1. We do not wish to split our conversations to obtain inter-speaker similarities, nor do we wish to drastically alter the agglomerative algorithm if it can be avoided. We need do neither if we simply set all possible inter-speaker similarities between speakers in two conversations equal to the similarity between the two conversations, so this is the approach we take.

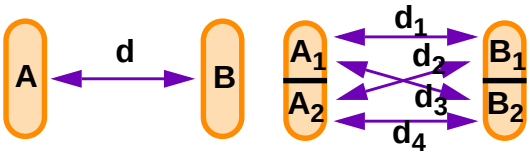

Figure 1: Similarities are computed only between conversations as shown at left. We initialize our algorithm with similarities between all pairs of speakers (two for each conversation) by setting $d_1 = d_2 = d_3 = d_4 = d$, as shown at right.

Agglomeration

With our multi-speaker agglomerative algorithm, we cannot allow agglomeration to proceed without constraints as it does in the standard algorithm. During the agglomeration process, we seek to enforce two rules: (1) each conversation must be placed in exactly two clusters and (2) no two conversations can have the same two speakers (no "repeat" conversations). The latter rule embodies a feature of our database that we take advantage of. Both rules can be adhered to by simply keeping track of clusters which we will not allow to merge. Conceptually, we can think of the similarity between such clusters as being zero, and therefore, the algorithm will never select them for merging.

With the initialization as described and the above rules in place, we agglomerate. Note that because not all merges are allowed, the tree that results from this two-speaker agglomerative algorithm will never produce a single root when it completes.

Tree Cutting

To produce clusters from a tree, the tree must be cut. Since each tree node represents a cluster of conversations, tree cutting is nothing more than the selection of a set of nodes. Our process for this selection, which we call "level cutting," involves horizontal cuts of the tree where the selected nodes are all the nodes immediately below the cut. (Leaves, here, are at the bottom of the tree, and the root is at the top.) Of course, we must perform the cutting without the benefit of knowing truth (which speakers are involved in which conversations).

Assessing Cuts

To choose the nodes which will serve as clusters we employ the BBN Metric defined in [3] as:

$$I_{BBN} = \sum_{i=1}^{N_c} n_i p_i - Q N_c \qquad (1)$$

Here, n_i is the number of conversations in cluster i, N_c is the total number of clusters and Q is a factor that can bias the tree cutting towards a small number of large clusters (by using a large Q) or towards a large number of small clusters

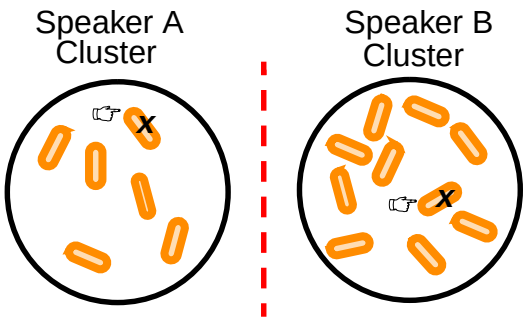

Figure 2: Conceptual illustration of conversations grouped into 2 clusters. Conversation *x* has near neighbors in both clusters.

(by using a small Q). We use $Q = 0.5$. The factor p_i, the purity of cluster i, captures the homogeneity of that cluster. If all conversations in the cluster involve the same speaker, then the purity is one.

To make the cut, we compute I_{BBN} for all sets of nodes (all possible level cuts), and then choose the set with the largest BBN value.

Estimating Purity

In a fair blind clustering exercise, we do not know the speakers in each of our conversations, and so we cannot know the true purity of clusters. To cut the tree, we must therefore estimate purity.

One such estimator, the *nearest neighbor* estimator, was proposed in [3]. To calculate purity for a cluster using this approach, we compute an *utterance purity* for every conversation in the cluster. If the number of conversations in the cluster of interest is n_i, then the utterance purity is that fraction of the conversation's nearest neighbors (as determined using the similarity measure) which are included among the n_i. Cluster purity is then the average of all the utterance purities.

For our two-speaker application, a small modification must be made to this procedure. Since each conversation contains two speakers A and B, then if our similarity measure is at all accurate, each conversation will have some near neighbors due to its A half and some near neighbors due to its B half. This situation is illustrated in Figure 2. When counting nearest neighbors for conversation x in cluster A, we must exclude from our counting any conversations that are near to x but located in B. Including such conversations improperly lowers our purity estimate. Note that unlike the nearest neighbor estimator for single-speaker data, our estimator requires a complete set of candidate clusters since we must have information about the "other half" of each conversation. We cannot estimate purity of a single cluster without knowledge of the other clusters.

4. APPLYING TRUTH

To evaluate the correctness of the clusters that arise from our tree cut, we must have truth. This allows us to calculate the true purity of each cluster and a "true" value for I_{BBN}. For convenience in comparing different clustering results, we convert this value to what we call the *clustering efficiency.*

True Purity

Knowing the true cluster memberships for each conversation, we can define true purity as

$$p_i = \sum_j \left(\frac{n_{ij}}{n_i}\right)^2 \qquad (2)$$

where n_{ij} is the number of conversations in cluster i that, in truth, come from speaker j. The sum runs over all speakers in the cluster. If all conversations come from the same speaker, then $n_{ij} = n_i$ and the purity is one. This is discussed in more detail in [3].

Implementing (2) is not entirely straightforward for the two-speaker case. Each conversation can only be counted once, however since each conversation has two speakers, we cannot be sure if a particular conversation should be counted due to its "speaker A" side or its "speaker B" side. In practice, for each conversation we choose the speaker which yields the largest value of p_i for that cluster. This "greedy" approach tends to overstate somewhat the true purity and hence the value of the BBN metric as well, but finding the correct speaker half for each conversation in each cluster is a difficult problem with no simple solution.

Clustering Efficiency

We would like to represent our value of I_{BBN} for a particular clustering relative to the best and worst that we could do. To achieve this, we calculate the clustering efficiency (CE) from [1]:

$$CE = \frac{I_{BBN} - F}{O - F} \qquad (3)$$

Here, F is the value of the BBN metric for a full-search (all singletons), and O is the BBN value for the optimum clustering. $F = N(1 - Q)$ and $O = N - N_c Q$ where N is the number of conversations.

5. RESULTS ON SWITCHBOARD SPEECH

To test this two-speaker agglomerative algorithm, 1101 two-minute conversations from all phases of the Switchboard corpus were produced by summing the conversation halves to create two-speaker conversations. The total number of speakers over the 1101 conversations was 505. No two speakers were involved in more than one

conversation, and each conversation contained exactly two speakers.

Table 1 shows results when employing the two-speaker agglomerative clustering to the similarity matrix calculated from these 1101 conversations. Also given are results using

Table 1: Clustering Results on Switchboard

Cluster Method	Split Method	No. of Clusters	I_{BBN}	CE
d*	Automatic	763	1587.9	0.58
d*	True	1031	1644.0	0.65
2-speaker agglom	No split	960	1430.2	0.39

d* [1]. Two different splitting methods were investigated: splitting using truth marks provided with the database and splitting using the automatic algorithm described in Section 1. As mentioned previously, automatic splitting hurts clustering efficiency, but automatic splitting followed by d* clustering is still superior to two-speaker agglomerative clustering on the similarity matrix.

In clustering the two-speaker conversations, two operations were performed. First, a similarity matrix was computed and then this was followed by the actual clustering. It is important to be able to assess the degree to which each of these parts is responsible for the low clustering efficiency.

One way to judge the discriminating power of a set of distances or similarities is by the detection error trade-off (DET) curve — essentially a receiver operator characteristic. Figure 3 shows a DET curve for our two-speaker similarities

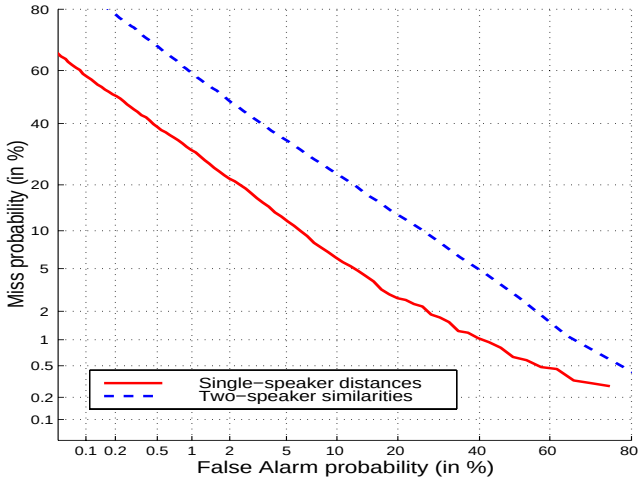

Figure 3: DET curves for two-speaker similarities and single-speaker distances. The single-speaker has a lower false alarm and miss rate at all thresholds, and therefore has much greater discriminating power.

calculated on the Switchboard 1101-conversation set and another curve for 1369 single-speaker Switchboard conversations. The DET curves show that the technique used for computing the single-speaker distances gives us far greater discrimination power than in the two-speaker case, and not surprisingly, using d* we routinely achieve much greater clustering efficiencies than we have with the two-speaker agglomerative approach.

6. CONCLUSIONS

We have described and tested an approach to clustering of two-speaker telephone conversations that does not require splitting of each conversation into its respective speaker halves. Though at present our result is not as good as when we split and cluster, our clustering algorithm may not be entirely to blame. We have shown that the method for computing inter-conversation similarities lacks the discriminating power of the analogous technique which we apply in the case of single-speaker telephone data.

An obvious course for future work to take is toward a better method to two-speaker similarity computation. We noted previously that the present similarities are not normalized for duration. Performing such a normalization, though it has its drawbacks (as noted), may lead to improved performance.

REFERENCES

[1] D. Reynolds, E. Singer, B. Carlson, G. O'Leary, J. McLaughlin and M. Zissman, "Blind Clustering of Speech Utterances Based on Speaker and Language Characteristics," *Int. Conf. Spoken Language Processing*, 1998. To appear.

[2] L. Wilcox, F. Chen, D. Kimber and V. Balasubramania, "Segmentation of Speech Using Speaker Identification," *Proc. ICASSP*, vol. 1, pp. 161-164, 1994.

[3] A. Solomonoff, A. Mielke, M. Schmidt and H. Gish, "Clustering Speakers by Their Voices," *Proc. ICASSP*, vol. 2, pp. 757-760, May, 1998.

[4] D.A. Reynolds, "Speaker Identification and Verification Using Gaussian Mixture Speaker Models," *Speech Communication*, vol. 17, pp. 91-108, August, 1995.

[5] D.A. Reynolds, "Comparison of Background Normalization Methods for Text-Independent Speaker Verification," *Eurospeech*, Sept., 1997.

[6] J.A. Hartigan, **Clustering Algorithms**, Wiley, New York, 1975.