A NEW DP-LIKE SPEAKER CLUSTERING ALGORITHM

Zhijian Ou, Zuoying Wang

Department of Electronic Engineering,
Tsinghua Univ., Beijing 100084, P.R.China
ozj@thsp.ee.tsinghua.edu.cn

ABSTRACT

In this paper we propose a new segment-synchronous speaker clustering algorithm based on the Bayesian Information Criterion (BIC), which is motivated by the Dynamic Programming (DP) idea. Compared with the commonly used agglomerative speaker clustering methods, the proposed algorithm is faster for lack of distance-matrix building and more reasonable as it avoids in some degree the simple irrevocable merging fashion. Moreover it facilitates online speaker clustering, which is important for real-time transcription applications (e.g., broadcast news, teleconferences etc.). In our experiments on 1997 Hub4 Mandarin broadcast news development data, unsupervised speaker adaptation with this DP-like clustering achieved 17.66% relative reduction in Character Error Rate (CER) from the baseline, as much as with the clustering by the true speaker identities.

1. INTRODUCTION

It is known that speaker adaptation can significantly improve the performance of large-vocabulary Automatic Speech Recognition (ASR) systems [1]. Speaker adaptation uses speech data from one speaker to adjust the parameters of the speaker-independent system towards the speaker-dependent values. As more adaptation data is used, the speaker adaptation becomes more effective. In the task of transcribing the real-world speech such as broadcast news, there are no speaker labels and the same speaker may appear multiple times in the long audio stream. It is therefore required that all the audio segments originating from a common speaker be clustered together. Speaker adaptation can then be applied on each speaker cluster, usually in such an unsupervised manner.

Various speaker clustering algorithms [3,4,5] have been proposed in the context of improving unsupervised speaker adaptation, which actually can all be categorized as the agglomerative clustering methods [2]. They distinguish themselves from each other by different segment-to-segment distance measures (e.g. Kullback-Liebler distance [5] or generalized likelihood ratio distance [3,4]), different cluster-to-cluster distance definition (e.g. maximum linkage [3]), and different criteria as to how to pick the desired clustering solution (e.g. by thresholding the distances [5] or BIC [3]). The agglomerative clustering algorithm often performs in three stages and suffers two weaknesses. In the first stage, it has to compute distances between each pair of segments, thus can define distances between clusters that are used to guide subsequent merging. Such kind of distance-matrix building is usually expensive, which is the first weakness. Then by starting with each segment in its own cluster and successively merging two "nearest" clusters, we can create a clustering solution tree. Once two segments have been merged they cannot subsequently be separated, which is the second weakness. The last stage is to pick the desired clustering solution according to some criterion.

In this paper, a new DP-like segment-synchronous speaker clustering algorithm is proposed to circumvent these weaknesses. It takes the segments to be clustered as a sequential input. In each step, as a new segment comes in, a group of clustering solutions of the already inputted segments, with different numbers of clusters in each solution, is optimally constructed, according to BIC, from the old group of clustering solutions obtained in the last step. Compared with the agglomerative speaker clustering algorithms [3,4,5], our algorithm is faster for lack of distance-matrix building and more reasonable as it avoids in some degree the simple irrevocable merging fashion. Moreover it facilitates online speaker clustering, which is important for real-time transcription applications (e.g., broadcast news, teleconferences etc.). Our experiments show that this speaker clustering algorithm improves unsupervised speaker adaptation as much as the clustering by the true speaker identities.

This paper is organized as follows. Section 2 describes the details of the speaker clustering algorithm and also compares our algorithm with other recent works. In section 3, some experimental results are provided to show the effectiveness of this algorithm. Finally the conclusions are drawn in section 4.

2. ALGORITHM DESCRIPTION

Let $S = \{s_i, i = 1, \cdots, I\}$ be the collection of audio segments we wish to cluster and each s_i represents a sequence of spectral vectors, i.e., the Cepstral vectors extracted from the i^{th} segment. A clustering solution with regard to the whole set S (or say, A clustering solution of S) is a partition $P_K^S = \{c_k, k = 1, \cdots, K\}$ of S, where K is the number of clusters. It is our task to find the best clustering solution with regard to the whole set S.

2.1. BIC Criterion

A clustering solution can be viewed as a kind of model description of the data set. So the Bayesian Information Criterion (BIC) [3], as a well-known model selection criterion, is appropriate here for picking the desired one among multiple candidate clustering solutions. To be specific let χ be the data set we are modeling, N be the size of the data set, M

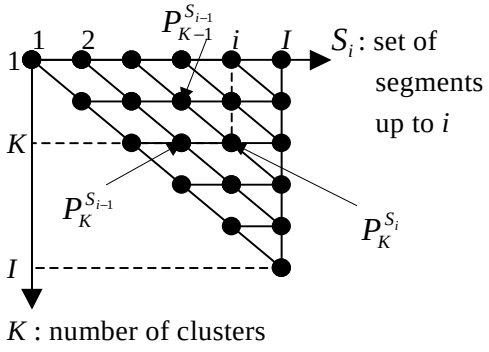

Fig. 1: Illustration of the forward recursion based on a lattice structure

be a candidate parametric model, $L(\chi, M)$ be the likelihood of χ generated by M, $\#(M)$ be the number of parameters in the model M. Then the BIC value of M is defined as

$$\mathrm{BIC}(M) = \log L(\chi, M) - \lambda \cdot \frac{1}{2} \cdot \#(M) \cdot \log N,$$

where λ is the penalty weight. The BIC criterion is to choose the model with the maximum BIC value.

To apply BIC in cluster analysis, we model each cluster c_k as a multivariate Gaussian distribution $N(\mu_{c_k}, \Sigma_{c_k})$, where μ_{c_k} can be estimated as the sample mean vector and Σ_{c_k} can be estimated as the sample covariance matrix. Let n_{s_i} be the number of frames in segment s_i, n_{c_k} be the number of frames in cluster c_k, i.e., $n_{c_k} = \sum_{i: s_i \in c_k} n_{s_i}$. Then one can show that

$$\mathrm{BIC}(P_K^S) = \sum_{k=1}^{K} \left\{ -\frac{1}{2} n_{c_k} \log \left| \Sigma_{c_k} \right| \right\} - \lambda \cdot \frac{1}{2} \cdot K \left[d + \frac{d(d+1)}{2} \right] \log N_{P_K^S},$$

where $N_{P_K^S} = \sum_{k=1}^{K} n_{c_k}$.

2.2. Segment-synchronous Clustering

Let $S_i = \{s_1, \cdots s_i\}$, $1 \le i \le I$, be the set of all the segments from s_1 up to s_i. Obviously $S_I = S$. A partition of S_i with K clusters is denoted by $P_K^{S_i}$, $1 \le K \le i$. Then the basic idea is that a good clustering solution $P_K^{S_i}$ of S_i can be derived from the good clustering solution $P_{K-1}^{S_{i-1}}$ and $P_K^{S_{i-1}}$ in an organized way.

With the set-theoretic operators ($\cup$: union, $\setminus$: difference), our proposed algorithm can be represented as follows.

I. Initialization: $P_1^{S_1} = \{\{s_1\}\}$ is readily available.

II. Forward Recursion:

$i = 2, \cdots, I$

$K = i, i-1, \cdots, 1$

Construct $P_K^{S_i}$ from active $P_{K-1}^{S_{i-1}}$ and active $P_K^{S_{i-1}}$.

Specifically, let $P' = P_{K-1}^{S_{i-1}} \cup \{\{s_i\}\}$,

$P_k'' = \left[P_K^{S_{i-1}} \setminus \{c_k\}\right] \cup \{c_k \cup \{s_i\}\}$,

$c_k \in P_K^{S_{i-1}}$, $k = 1, \cdots, K$,

then $P_K^{S_i} = \arg\max_{P \in \{P', P_k'', k=1,\cdots,K\}} \mathrm{BIC}(P)$. (*)

Set the activity status of $P_K^{S_i}$, $K = i, \cdots, 1$, according to their BIC values.

III. Termination:

Do $\max_{1 \le K \le I} \mathrm{BIC}(P_K^{S_I})$, $\hat{K} = \arg\max_{1 \le K \le I} \mathrm{BIC}(P_K^{S_I})$ and choose $P_{\hat{K}}^{S_I}$ as the final best clustering solution.

The operation of (*) is performed for all possible number of clusters (i.e., $K = i, \cdots, 1$) with regard to a given set S_i; the operation is then iterated for $i = 2, \cdots, I$, i.e., in a segment-by-segment manner. The clustering process goes on as each segment comes in, so named segment-synchronous clustering, and in effect is based on the lattice structure shown in Fig. 1. The forward recursion is the heart. To further clarify the meaning of (*), suppose that a new segment s_i comes in and we know that the "best" clustering solution of S_{i-1} with cluster number $K-1$ is $P_{K-1}^{S_{i-1}}$ and the "best" clustering solution of S_{i-1} with cluster number K is $P_K^{S_{i-1}}$. Then intuitively we can simply add a cluster comprised of only s_i, i.e., $\{s_i\}$, to $P_{K-1}^{S_{i-1}}$, or we can separately assign s_i to each cluster in $P_K^{S_{i-1}}$, and then choose the one that maximizes the BIC value. This is just what (*) does.

To sum up, we take the segments to be clustered as a sequential input. In the i^{th} step, as the i^{th} segment s_i comes in, we construct a group of partitions of S_i, ranging from a single big cluster (i.e., $P_1^{S_i} = \{s_1, \cdots s_i\}$) to i clusters each containing a single segment (i.e., $P_i^{S_i} = \{\{s_1\}, \cdots \{s_i\}\}$), by deriving partitions of S_{i-1} which we have obtained in the $(i-1)^{\text{th}}$ step to optimally include the i^{th} segment.

2.3. Discussion

Various speaker clustering algorithms [3,4,5] have been proposed, which actually can all be categorized as the agglomerative clustering algorithms as follows.

I. Distance-matrix Building: For each pair of segments s_i and s_j, compute the distance $d(s_i, s_j)$.

II. Initialization: $P_I^{S_I} = \{\{s_1\}, \cdots \{s_I\}\}$ is readily available.

III. Successively Merging:

$K = I, \cdots, 2$

Merge two "nearest" clusters. Specifically,

Let $(k_1, k_2) = \arg\min_{1 \le k_1 < k_2 \le K} d(c_{k_1}, c_{k_2})$ where $d(c_{k_1}, c_{k_2})$ denotes the distance between cluster c_{k_1} and cluster c_{k_2}, then $P_{K-1}^{S_I} = \left[P_K^{S_I} \setminus \{c_{k_1}, c_{k_2}\}\right] \cup \{c_{k_1} \cup c_{k_2}\}$.

IV. Termination:

Pick the desired $P_{\hat{K}}^{S_I}$ from the clustering solution tree $P_I^{S_I}, P_{I-1}^{S_I}, \cdots, P_1^{S_I}$.

It can be seen from the algorithm description above that our algorithm is very different from traditional agglomerative clustering algorithms. In each step of the agglomerative algorithm it merges two "nearest" clusters, that is, segments in these two clusters are bound together and no longer can be separated. All the clustering solutions subsequently created are therefore subject to such limit of irrevocable merging. Merging in our algorithm is local and influences only next two clustering solutions, which allows for other clustering possibilities through a group of clustering solutions in each step (in Fig. 1 each column stands for a group of clustering solutions each with different numbers of clusters). More information is preserved.

To examine the computation involved in the DP-like algorithm, we see that the number of determinant calculations is about $O(I^3)$, or precisely, $I + \sum_{i=2}^{I}(1+2+\cdots+(i-1))$. But we find that for a given S_i, the operation of (*) is actually not necessarily performed for all possible $K = i, \cdots, 1$. Usually $I >> K^*$, the actual number of speakers. So at the end of the loop of K, we can keep only top W clustering solutions of $P_K^{S_i}$'s active, according to their BIC values, and set others to be null, where W is the active width. In the following recursive construction of new clustering solutions, old null clustering solutions are ignored. Incorporating the above technique in the DP-like algorithm, we reduce the number of determinant calculations to about $O(WK^*I)$.

The speaker clustering algorithm proposed by Chen et al. [3] is a typical agglomerative clustering algorithm with maximum linkage and gives state-of-the-art performance. It uses BIC as the termination criterion (i.e., two clusters can be merged only if the merging increases the BIC value). It is worthwhile comparing computational aspects of our algorithm to those of that algorithm. It can be shown that the number of determinant calculations in that algorithm is about $O(0.5 \cdot I(I+1) + (I - K^*))$, which is usually greater than $O(WK^*I)$. The term $0.5 \cdot I(I+1)$ accounts for the distance-matrix building required by the agglomerative clustering with the generalized likelihood ratio [3] as the distance measure. We take that algorithm as an example of the various agglomerative clustering algorithms and give a more detailed comparison experimentally in section 3.

In general the agglomerative speaker clustering algorithms work in an offline manner. Only after all the audio segments are obtained, the system starts clustering. As shown above, our segment-synchronous clustering algorithm naturally facilitates clustering the segments online. The system need not wait for all the segments to come in. At each moment it always gives a group of clustering solutions of all the already inputted segments each with different numbers of clusters, from which the desired one can then be picked, according to BIC, to guide unsupervised speaker adaptation if necessary. Such kind of online incremental speaker clustering and unsupervised speaker adaptation is important for real-time transcription applications.

3. EXPERIMENTAL RESULTS

We carried out experiments to assess the effectiveness of the clustering algorithm on 1997 Hub4 Mandarin broadcast news development data. All together there are 58 files, where 53 files are used to train the acoustic model, the duration distribution based HMM (DDBHMM)[6]. The language model is built from 93, 94 People's Daily and the above 97 Hub4 data. The remaining 5 files are used as test data. For comparison the algorithm proposed in [3] was also implemented, which will be referred to as the term "agglomerative clusterer" below. We experimented with several feasible λ, W, and finally chose $\lambda = 3$ and $W = 10$, which were used throughout all the following experiments. Starting from a baseline system without adaptation, we employed respectively the clustering by the true speaker identities, the agglomerative clusterer and the DP-like clusterer to guide subsequent unsupervised speaker adaptation that is based on Maximum Likelihood Linear Regression (MLLR) [1].

The purity of a cluster is defined as the ratio between the number of segments by the dominating speaker in that cluster and the total number of segments in that cluster. It can be seen from Table 2 and Fig. 2 that our algorithm results in clusters with both appropriate number of clusters and high purity. Note that for space consideration, purity result for file 1 only is shown here.

Since the clustering is to be used for subsequent speaker adaptation, a natural evaluation of clustering is based on how well the clusters perform in adaptation. It can be seen from Table 1 that for different files the DP-like clusterer performs as well as the agglomerative clusterer consistently. On the average the relative Character Error Rate (CER) reduction of adaptation with the DP-like clustering is 17.66% from the baseline, compared to 17.32% with the agglomerative clusterer. Both perform slightly better than the clustering by the true speaker identities, which is not too surprising. Since there is little speech for some speakers in the files, it might actually help reduce CER by merging speakers in the same cluster if their acoustic characteristics are similar, where the

Experiment	1	2	3	4	5	average
Baseline without adaptation	32.2%	32.3%	41.6%	25.9%	15.2%	29.44%
By true speaker identities	28.0%	26.8%	36.0%	21.4%	9.9%	24.42%
Aggl. clusterer	26.9%	26.8%	36.2%	21.6%	10.2%	24.34%
DP-like clusterer	27.5%	26.8%	35.8%	21.1%	10.0%	24.24%

Table 1: % CER with different clusterers on 5 test files

Experiment	1	2	3	4	5
Number of spkrs.	21	17	13	11	3
Aggl. clusterer	50	46	10	13	4
DP-like clusterer	17	16	9	10	2

Table 2: Number of clusters generated by different clusterers on 5 test files

Experiment	1	2	3	4	5
Aggl. clusterer	175477	163831	84244	32882	16831
DP-like clusterer	49048	32957	20664	12504	6514

Table 3: Actual number of determinant calculations by different clusterers

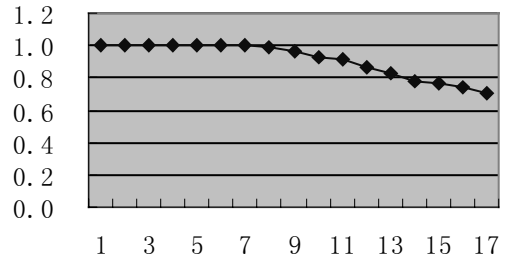

Fig. 2: Clustering Purities (Y axis) for the 17 clusters (X axis) chosen by our algorithm, applied to file 1 where the true number of speakers is 21.

MLLR transforms can be more robustly estimated. On the other hand, divisions of speech from the same physical speaker coming from significantly different channel and/or background conditions into two clusters would also be beneficial to reduce CER.

Moreover the actual number of determinant calculations of both algorithms is counted and summarized in Table 3. It is clear that, owing to the lack of distance-matrix building for the segments, the computational load of the DP-like clustering is significantly less than that of the agglomerative clusterer. Hence the recognition gain achieved by the DP-like clustering is due to its more effective clustering fashion.

4. CONCLUSIONS

In this paper, a new DP-like segment-synchronous speaker clustering algorithm is proposed to circumvent some weaknesses of the agglomerative speaker clustering algorithms. Experiments show that it improves CER as much as the clustering by the true speaker identities in unsupervised speaker adaptation. Clearly this algorithm provides a rather general cluster analysis framework, not restricted to speaker adaptation, and BIC is not the only criterion that can be used here.

5. REFERENCES

[1] C.J. Leggetter and P.C. Woodland, “Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density HMMs”, Computer Speech and Language, Vol.9, pp. 171-185, 1995

[2] B.S. Everitt, “Cluster Analysis”, Halsted Press, New York, Third edition, 1993

[3] S. Chen and P.S. Gopalakrishnan, “Clustering via the Bayesian Information Criterion with Applications in Speech Recognition”, Proc. ICASSP 1998, pp. 645-648

[4] H. Jin, F. Kubala and R. Schwartz, “Automatic Speaker Clustering”, Proceedings of the DARPA Speech Recognition Workshop, pp. 108-111, 1997

[5] M. Siegler, U. Jain, B. Ray and R. Stern, “Automatic Segmentation, Classification and Clustering of Broadcast News Audio”, Proceedings of the DARPA Speech Recognition Workshop, pp. 97-99, 1997

[6] Zuoying Wang, “Inhomogeneous HMM for Speech Recognition and THED Recognition and Understanding System”, Telecommunication Science, Vol.9, No.4, pp. 31-36, 1993. (in Chinese)