A NOVEL SPEAKER CLUSTERING ALGORITHM VIA SUPERVISED AFFINITY PROPAGATION

Xiang Zhang, Jie Gao, Ping Lu, Yonghong Yan

ThinkIT Speech Lab, Institute of Acoustics, Chinese Academy of Sciences, Beijing, P.R.China
{xzhang, jgao, plu, yonghong.yan}@hccl.ioa.ac.cn

ABSTRACT

This paper addresses the problem of speaker clustering in telephone conversations. Recently, a new clustering algorithm named affinity propagation (AP) is proposed. It exhibits fast execution speed and finds clusters with low error. However, AP is an unsupervised approach which may make the resulting number of clusters different from the actual one. This deteriorates the speaker purity dramatically. This paper proposes a modified method named supervised affinity propagation (SAP), which automatically reruns the AP procedure to make the final number of clusters converge to the specified number. Experiments are carried out to compare SAP with traditional k-means and agglomerative hierarchical clustering on 4-hour summed channel conversations in the NIST 2004 Speaker Recognition Evaluation. Experiment results show that the SAP method leads to a noticeable speaker purity improvement with slight cluster purity decrease compared with AP.

Index Terms— speaker clustering, affinity propagation, supervised affinity propagation, generalized likelihood ratio

1. INTRODUCTION

In the recent past, interests and needs in speech recognition community (by enabling the use of speaker dependent systems) have provided a major motivation for the research on speaker clustering, which refers to the task of classifying the speech utterances into clusters such that each cluster contains speech from one speaker and also speech from the same speaker is grouped into the same cluster[1, 2, 3, 4, 5, 6]. Speaker clustering is usually based on either the BIC criterion or on Cross Likelihood Ratio (CLR) [7, 8]. A thorough overview of available speaker clustering algorithms is given in [9].

Current approaches for speaker clustering can be classified into two categories: unsupervised and supervised techniques. The main application considered in this research is to cluster the speech segments in telephone conversations with known number of speakers, which is a supervised speaker clustering problem. The conventional k-means and agglomerative (bottom-up,clumping) hierarchical clustering (AHC) are two widely used clustering methods. K-means [10] begins with an initial set of randomly selected c cluster centers, where c is the desired number of clusters. Then it reruns several times to update the centers until the cluster centers do not change. This method is quite sensitive to the initial selection of centers, which affects the number of iterations and the precision of speaker clustering. In this paper, AHC uses the number of speakers as stopping criterion. The AHC procedure [4, 9] starts with initializing leaf clusters of tree with speech utterances and calculating pair-wise distances. Then it continues to aggregate clusters together and update distances of clusters to the new cluster. The procedure terminates when the specified number of clusters has been obtained. This method does not require that all utterances within a cluster be similar to a single center and is thus not well-suited to some tasks.

Affinity propagation [11] is a clustering method proposed recently, which has been used to cluster images of face, identify representative sentences, detect genes, and so on [12]. In this paper, we introduce it into our system to cluster speech segments. Affinity propagation assumes that all the speech segments are potential centers. By viewing each segment as a node in a network, affinity propagation recursively transmits real-valued messages along edges of the network until a good set of centers and corresponding clusters emerges. As described later, messages are updated on the basis of simple formulas during the procedure with precomputed similarities. Affinity propagation is an unsupervised clustering method, which is suitable for the situations of unknown number of speakers. However, our experiment shows that when the number of speakers is given, adopting affinity propagation for speaker clustering may generate undesired number of clusters and low speaker purity.

This paper proposes a modified method named supervised affinity propagation (SAP). It is an adaptive approach based on affinity propagation, which automatically reruns the affinity propagation procedure to make the resulting number of clusters converge to the desired number of speakers. This method has fast processing speed and yields high precision. The speaker purity generated by SAP is improved significantly with slight decrease in execution speed and cluster purity compared with affinity propagation.

The rest of this paper is organized as follows: In Section 2, affinity propagation algorithm is introduced into speaker clustering. In Section 3, an adaptive approach named SAP is proposed to improve the system performance with specified number of clusters. In Section 4, we present the main evaluation metrics for speaker clustering and give the comparative results. Finally, the conclusion is given in Section 5.

2. SPEAKER CLUSTERING VIA AP

Recently, a new clustering approach named affinity propagation (AP) is proposed, which has fast processing speed and can avoid many of the poor solutions caused by unlucky initializations and hard decisions by simultaneously considering all the data points as candidate centers and gradually identifying clusters. Thus, we introduce AP for speaker clustering in telephone conversations.

This work is partially supported by MOST (973 program, 2004CB318106), the National Natural Science Foundation of China (10574140, 60535030), the National High Technology Research and Development Program of China (863 program, 2006AA01010, 2006AA01Z195).

The *similarity* $s(i,k)$, where $i \neq k$, *preference* $s(k,k)$, *responsibility* $r(i,k)$ and *availability* $a(i,k)$ are the four main elements in AP. AP takes a collection of real-valued similarities between speech segments as input, where the similarity $s(i,k)$ indicates how well the segment k is suited to be the center for the segment i. The preference $s(k,k)$ is a real number for each segment k. The segments with larger values of $s(k,k)$ are more likely to be chosen as centers. If a priori, all the segments are equally suitable as centers, the preferences should be set to a common value. The responsibility $r(i,k)$ reflects the accumulated evidence for how well-suited segment k is to serve as the center for segment i, taking into account other potential centers for segment i. The availability $a(i,k)$ reflects the accumulated evidence for how appropriate it would be for segment i to choose segment k as its center, taking into account the support from other segments that segment k should be a center.

In our AP speaker clustering, each similarity is set to the negative generalized likelihood ratio (GLR):

$$s(i,k) = -d_{GLR}(x_i, x_k),\ i \neq k \tag{1}$$

where, x_i and x_k are the speech feature vectors of the two segments, which can be modeled with two Gaussian models $N(\mu_{x_i}, \sum_{x_i})$ and $N(\mu_{x_k}, \sum_{x_k})$. $d_{GLR}(x_i, x_k)$ is the GLR distance between x_i and x_k, which is defined as follows [2]:

$$d_{GLR}(x_i, x_k) = \frac{L(x_i; \mu_{x_i}, \sum_{x_i}) \cdot L(x_k; \mu_{x_k}, \sum_{x_k})}{L(y; \mu_y, \sum_y)} \tag{2}$$

where, $L(*)$ is the likelihood function and y is the union of x_i and x_k which indicates the concatenation of both feature vectors.

The suitable input preferences are very important to influence the final number of clusters. The larger the values of preferences, the more clusters will be produced. We set all the preferences to the same value – the median of total input similarities as mentioned in [11]. Responsibility and availability are two kinds of message exchanged between speech segments, which are iteratively updated by the formulas (3) (4) (5), reflecting the affinity of segments. They are computed as follows:

$$r(i,k) = s(i,k) - \max_{j:j\neq k}[s(i,j) + a(i,j)] \tag{3}$$

For $a(i,k)$, if $k = i$,

$$a(i,k) = \sum_{i':i'\neq k} \max[0, r(i^{'}, k)] \tag{4}$$

If $k \neq i$,

$$a(i,k) = \min\{0, r(k,k) + \sum_{i':i'\notin\{i,k\}} \max[0, r(i^{'}, k)]\} \tag{5}$$

For the first iteration, the availabilities are initialized to zero.

AP combines the responsibilities and availabilities to control the center decisions. For segment i, the segment k which maximizes $r(i,k) + a(i,k)$ either identifies the segment i as a center if $k = i$, or identifies the segment that is the center for segment i if $k \neq i$. The whole AP procedure terminates after a fixed number of iterations or after the center decisions stay unchanged for some number of iterations. The AP speaker clustering procedure goes as Algorithm 1, where N denotes the number of segments and M is the maximum number of iterations.

The procedure of the algorithm shows that the similarities are computed only once and just the two kinds of message need to be computed based on the known similarities.

Algorithm 1 (Affinity propagation speaker clustering)

begin initialize

$$N, M, a(i,k) \leftarrow 0, i = 1:N, k = 1:N$$

$$s(i,k) \leftarrow -d_{GLR}(x_i, x_k), i = 1:N, k = 1:N, i \neq k$$

$$s(k,k) \leftarrow \underset{i=1:N, j=1:N, i\neq j}{\text{median}}[s(i,j)], k = 1:N$$

while $M \neq 0$ **do**
  update $a(i,k)$ and $r(i,k)$ using equations (3)(4)(5)
  For each x_i, search the segment k which maximizes $r(i,k) + a(i,k)$ to identify whether segment i is a center or which segment is the center for it.
  $M \leftarrow M - 1$
  if all the $r(i,k) + a(i,k)$ do not change for a fixed number of iterations **then**
    goto step 10
  end if
end while
end

3. SPEAKER CLUSTERING VIA SAP

3.1. Motivation of SAP

When the number of speakers (clusters) is given, affinity propagation would not be well-suited to speaker clustering. Our experiment shows that the number of the clusters produced by AP is often different from the expected number. This deteriorates the speaker purity dramatically due to the speech frames of one speaker assigned to extra clusters. The problem lies in the fact that AP is an unsupervised method that can not utilize the prior knowledge of the given number of speakers. In order to overcome the above limitation, this paper proposes a supervised speaker clustering approach named supervised affinity propagation (SAP), which can take the number of speakers as input. Experiment results show that this method improves the speaker purity significantly compared to AP.

3.2. SAP Speaker Clustering

As reported in [11, 12], the number of clusters in AP is influenced by the values of the input preferences. By setting preferences to appropriate values, the expected number of clusters through AP can be achieved. This motivates the following iterative clustering algorithm (SAP) with adaptive preferences.

Assuming that C_{exp} is the expected number of clusters, n is the current number of iterations in SAP and C_n is the number of clusters in the nth iteration. The SAP algorithm goes as Algorithm 2.

The equation (6) is used to modify the preferences, as long as the resulting number of clusters is different from the expected one. If the number of clusters is larger than the desired number, the preferences would be decreased. If smaller, the preferences would be increased.

$$s_{n+1}(k,k) = s_n(k,k) - step * (C_n - C_{exp}) \tag{6}$$

Where, $s_n(k,k)$ and $s_{n+1}(k,k)$ are the preferences of nth and $(n+1)th$ iteration, $step$ is the adaptive factor controlling the speed of convergence. The value of adaptive factor should be positive and can be determined empirically. In our experiment, we set the value to 20. The first minus sign in the updated equation assures the final cluster number C_n can converge to the desired cluster number C_{exp} with

Algorithm 2 (SAP)

begin initialize

$$n \leftarrow 0, C_0 \leftarrow C_{exp}, step$$

$$a(i,k) \leftarrow 0, i = 1:N, k = 1:N$$

$$s(i,k) \leftarrow -d_{GLR}(x_i, x_k), i = 1:N, k = 1:N, i \neq k$$

$$s_n(k,k) \leftarrow \underset{i=1:N, j=1:N, i \neq j}{\text{median}}[s(i,j)], k = 1:N$$

while $C_n \neq C_{exp}$ **or** $n = 0$ **do**
  $n \leftarrow n + 1$
  execute AP procedure (all the stpes in algorithm 1 without step 1)
  get the number of clusters: C_n
  if $C_n = C_{exp}$ **then**
    get cluster results
    goto step 13
  else
    update preferences using equation (6)
  end if
end while
end

an appropriate adaptive factor, as the resulting number of clusters decreases with the values of preferences.

In the proposed SAP algorithm, the similarities between segments are computed only once. Only the preferences need to be calculated after each iteration of SAP. The equations (3)(4)(5) in AP procedure and the updated equation (6) show that the computational complexity of SAP procedure increases not too much compared to AP, and our experiment results show that the speaker purity of SAP is improved significantly with slight cluster purity decrease.

4. EXPERIMENT RESULTS AND EVALUATION

Our experiments are all based on a hand labeled test set of the NIST 2004 Speaker Recognition Evaluation. The test set contains 4-hour telephone conversations, each of which contains two speakers. Speech features of 14 line spectrum pair (LSP) are extracted from these data for every 20-ms Hamming-windowed frame with 10-ms frame shifts. The experiments for AP and the proposed SAP are carried out on the test set, and they are compared with those for the conventional AHC and k-means, which are implemented respectively as described in [9] and [10]. All the algorithms use GLR to calculate the distance. In both AP and SAP, each similarity is set to a negative GLR distance. The preferences of AP are set to the median of similarities. The initial preferences of SAP are also set to the median of similarities.

4.1. Evaluation Metrics

We evaluate our experiments with execution time and other two commonly used criteria [4, 13, 14]: cluster purity (cp) and speaker purity (sp). Cluster purity is a quantity which describes to what extent all speech frames in the cluster come from the same speaker. Speaker purity is a quantity which reflects the percentage of frames of speaker belonging to dominant cluster.

First, we assume k is the total number of speakers, c is the final number of clusters, and n_{ij} denotes the number of speech frames in cluster i spoken by speaker j. Then the cluster purity and speaker purity are defined as:

$$cp = \sum_{i=1}^{c} \max_{j \in [1:k]}(n_{ij}) / \sum_{i=1}^{c}\sum_{j=1}^{k} n_{ij} \quad (7)$$

$$sp = \sum_{j=1}^{k} \max_{i \in [1:c]}(n_{ij}) / \sum_{i=1}^{c}\sum_{j=1}^{k} n_{ij} \quad (8)$$

4.2. Experiment results

The traditional k-means (the initialized number of seeds is 2, and the seeds are selected randomly), AHC (bottom-up, using the number of speakers as stopping criterion), AP and the proposed SAP are implemented respectively. Table 1 lists the AP results for conversations which have different numbers of speech segments. These conversations used are all chosen from the test set. Table 2 displays the corresponding performance of the four algorithms. The parameter t denotes the whole run-time based on our test set, N denotes the number of segments, and c is the resulting number of clusters.

conversation	N	c	cp (%)	sp (%)
1	14	2	99.3	99.3
2	29	2	94.4	94.4
3	40	3	98.8	94.9
4	48	4	98.4	90.2
5	58	5	91.5	68.4
6	65	4	99.7	77.5
7	77	5	96.1	92.2
8	94	6	98.6	44.8
9	109	7	93.4	51.0
10	144	8	97.8	46.8

Table 1. The AP results for different conversations

From Table 1, it can be observed that the resulting numbers of clusters through AP procedure for many conversations are different from the actual numbers of speakers, which are all equal to 2 in our experiments. We can see that the results of cluster purity for most conversations are excellent. However, the results of speaker purity for some conversations are not satisfying, especially when the number of speech segments is large. This is mainly caused by the fact that AP is an unsupervised algorithm which can not take advantage of the known number of speakers. This sometimes makes the resulting number of clusters different from the actual one and deteriorates the speaker purity dramatically. Therefore, when the number of speakers is given, a supervised clustering method used to improve the performance of speaker purity is needed.

	AHC	k-means	AP	SAP
t (ms)	75150	55350	71230	73120
cp (%)	90.1	92.6	94.5	92.4
sp (%)	91.6	92.6	84.4	95.2

Table 2. The performance via AHC, k-means, AP and SAP

Table 2 shows the cluster purity and speaker purity of AHC, k-means, AP and SAP on the whole test set. It can also be seen that AP generates the highest cluster purity, but the speaker purity of AP

is the lowest among the four methods. SAP leads to about 10.8% absolute improvement in speaker purity with about 2.1% decrease in cluster purity compared with AP. The facts show that the speaker purity of SAP is superior to that of AP when the number of speakers in the conversation is given, and the cluster purity of SAP decreases only a little compared with AP.

The results in Table 2 also show that SAP is superior to AHC in both cluster and speaker purity, and it generates better speaker purity than k-means. SAP achieves about 2.3% and 3.6% absolute improvements in cluster purity and speaker purity respectively compared with AHC and gets about 2.6% absolute improvement in speaker purity with almost the same cluster purity compared with k-means.

All the facts show that SAP can be well used in telephone speaker clustering task with known number of speakers. This is mainly because the AP procedure in SAP framework simultaneously considers all the segments as candidate centers and gradually identifies clusters, such that it is able to avoid some poor solutions caused by unlucky initializations and hard decisions.

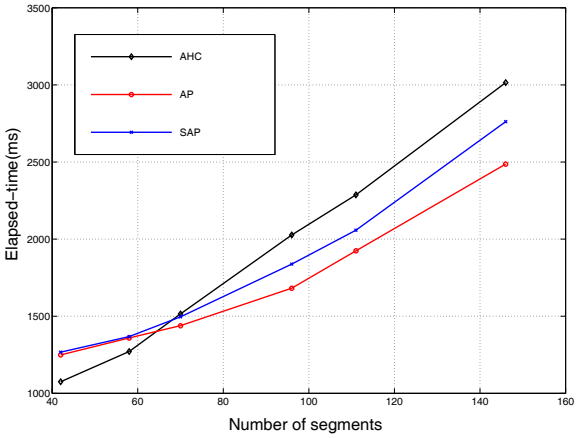

Fig. 1. The time cost of AHC, AP, and SAP

4.3. Run-time efficiency

From Table 2, it can be seen that AHC takes more execution time on the whole test set than AP and SAP, and k-means takes the least execution time. AHC begins with the calculation of the distances between clusters and needs to update the distances between the clusters and the new cluster. AP and SAP take a different approach, the similarities in which are computed only once. Then the procedures of both two methods go on with known similarities. However, k-means only needs iterative computation of the distances between each speech segment and the cluster centers. In practice, both the number of centers and the number of iterations are generally much less than the number of segments. We can see that AP and SAP have faster speed than AHC, and k-means usually has the fastest processing speed in the four methods.

Figure 1 provides the time cost of speaker clustering through AHC, AP, and SAP with different numbers of segments. The execution time of k-means is affected by the choice of initial cluster centers which influences the number of iterations, so the execution time of k-means is not given in Figure 1. However, k-means generally has the fastest speed in the four algorithms. We run the test on a series of data sets which contain from 40 to 150 segments. It can be observed that the execution time of the three algorithms increases with the number of speech segments, but the execution time of both AP and SAP is less than AHC when the number of segments is larger than 70, and our experiment results show that the run-time on the whole test set of both AP and SAP is less than AHC.

5. CONCLUSION

In this paper, we introduce the affinity propagation algorithm into our speaker clustering system and present a supervised speaker clustering approach named SAP which aims at processing real-world media with known number of speakers. Affinity propagation is an effective method which is suitable for the situation of unknown number of speakers. The proposed SAP approach, which is an improved approach based on affinity propagation, works with given number of speakers. We utilize SAP to cluster speech segments in telephone conversations, each of which has two speakers and the experiment results show that it can be well applied to the task of supervised speaker clustering.

6. REFERENCES

[1] D. Liu and F. Kubala, "Online speaker clustering," *Proc. ICASSP'03*, vol. 1, pp. 333–336, 2003.

[2] T. Stadelmann and B. Freisleben, "Fast and robust speaker clustering using the earth mover's distance and mixmax models," *Proc. ICASSP'06*, vol. 1, pp. 989–992, 2006.

[3] H. Jin, F. Kubala, and R. Schwartz, "Automatic speaker clustering," *Proc. DARPA Speech Recognition Workshop*, pp. 108–111, 1997.

[4] W. Wang, P. Lv, Q.W. Zhao, and Y. Yan, "A Decision-Tree-Based Online Speaker Clustering," *LECTURE NOTES IN COMPUTER SCIENCE*, pp. 555–562, 2007.

[5] A.N. Iyer, U.O. Ofoegbu, R.E. Yantorno, and B.Y. Smolenski, "Blind speaker clustering," *Proc. ISPACS'06*, pp. 343–346, 2006.

[6] J. Ajmera and C. Wooters, "A robust speaker clustering algorithm," *IEEE ASRU Workshop*, pp. 411–416, 2003.

[7] C. Barras, X. Zhu, S. Meignier, and J.L. Gauvain, "Improving speaker diarizaton," *Proc. DARPA RT04*, 2004.

[8] H. Aronowitz, "Trainable speaker diarization," *Proc. INTERSPEECH'07*, pp. 1861–1864, 2007.

[9] D.A. Reynolds and P. Torres-Carrasquillo, "Approaches and applications of audio diarization," *Proc. ICASSP'05*, vol. 5, pp. 953–956, 2005.

[10] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*, Wiley-Interscience, 2000.

[11] B.J. Frey and D. Dueck, "Clustering by Passing Messages Between Data Points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[12] B.J. Frey and D. Dueck, "Supporting online material for clustering by passing messages between data points," 2007.

[13] A. Solomonoff, A. Mielke, M. Schmidt, and H. Gish, "Clustering speakers by their voices," *Proc. ICASSP'98*, pp. 757–760, 1998.

[14] J. Ajmera, H. Bourlard, I. Lapidot, and I. McCowan, "Unknown-multiple speaker clustering using hmm," *Proc. ICSLP'02*, pp. 573–576, 2002.