Computer Science and Application 计算机科学与应用, 2020, 10(8), 1464-1470
Published Online August 2020 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2020.108153

Hans汉斯

Improvement of Fast HAC Clustering Algorithm and Application to Unsupervised Speech Segmentation

Zhanjiang Wei, Yu Liang

School of Software, Yunnan University, Kunming Yunnan
Email: 543819610@qq.com

Received: Jul. 26th, 2020; accepted: Aug. 10th, 2020; published: Aug. 17th, 2020

Abstract

HAC is a commonly used clustering method. According to the close relationship between phonemes and continuous time in speech features, the purpose of this paper is to improve the HAC fast algorithm to improve the unsupervised segmentation of speech signals to similar phoneme units. The algorithm is based on the fact that the similarity of the same segment feature is higher than that of the cross-segment feature. The similarity of features is to calculate the Euclidean distance between adjacent features to obtain the adjacent distance double-linked list of input speech features. Each node in the linked list is composed of the distance of adjacent speech features and pointers pointing to the adjacent nodes before and after. The algorithm also traverses the linked list of adjacent distance nodes, finds the minimum distance, combines similar adjacent features, and iterates to the last class or satisfies a certain threshold. The whole process is completed completely without supervision. This method is better than the fast HAC algorithm. Compared with the fast HAC algorithm, it can improve the clustering speed by more than 65 times, save more memory space, and can be applied to zero-resource speech segmentation.

Keywords

Unsupervised, Phoneme, HAC Algorithm, Speech Segmentation, Adjacent

快速HAC聚类算法的改进及应用于无监督语音分割

韦占江，梁　宇

云南大学软件学院，云南　昆明
Email: 543819610@qq.com

文章引用: 韦占江, 梁宇. 快速 HAC 聚类算法的改进及应用于无监督语音分割[J]. 计算机科学与应用, 2020, 10(8): 1464-1470. DOI: 10.12677/csa.2020.108153

收稿日期：2020年7月26日；录用日期：2020年8月10日；发布日期：2020年8月17日

摘　要

HAC是一种常用的聚类方法。本文的目的是根据语音特征中的音素与连续时间的紧密关系，改进HAC快速算法提高无监督分割语音信号到类似音素单位。该算法是基于同一段特征相似度高于跨段特征的相似度。特征的相似度是通过计算相邻特征间的欧式距离，来得到输入语音特征相邻的距离双链表，链表中的每个节点由语音相邻特征的距离和指向前后相邻节点的指针组成。该算法也是通过遍历相邻距离节点链表，查找最小距离后，对相似的相邻特征进行合并，并重复迭代至最后一个类或满足某个阀值。整个过程完全基于无监督下完成，该方法优于快速HAC算法，与快速HAC算法相比能提升65倍以上的聚类速度，节约更多的内存空间，可应用于零资源的语音分割。

关键词

无监督，音素，HAC算法，语音分割，相邻

1. 引言

人工智能对输入的未知事物进行分类判别，很大程度依据聚类将数据点按照一定规则进行分群[1]。根据距离或相似度对数据点聚类，是研究诸多自然和社会问题的有力工具。而在各种聚类算法中，分层聚类具有特别的优势[2]。聚类是一个把数据对象集划分成多个组或簇的过程，使得簇内的对象具有很高的相似性[2]。但与其他簇中的对象很不相似。数据间的距离通常使用距离度量。数据对象的簇可以看成隐含的类[3]。在这种意义下，聚类有时又称自动分类。聚类过程中可自动地发现这些分组，这是聚类分类的突出优点。也被称为无监督学习，因为在没有标签信息情况下，可实现自动分类。由于这种原因，聚类是通过观察学习，而不是通过有标签的示例学习。

语音信号可视为基本音素单元的序列。现今很多自动语音识别系统依赖于这些基本单元准确识别。当前成熟的百度搜索也是基于文字的检索，而基于语音对语音的搜索有巨大的市场需求。但无监督的语音研究困难重重，语音分割是声学片段建模中至关重要的初始步骤，该步骤在音频搜索和无监督声学建模得到了广泛应用。以语音分割成基本单位涉及声道系统从一种状态转换到另一种状态的时刻。但是，过渡并不是突然发生的，而是连续发生的[4] [5] [6] [7]。

通常，聚类的方法有很多种常见的有 K-means [8] [9]、层次聚类(Hierarchical clustering)、谱聚类(Spectral Clustering)、GMM 等方法[8]。其中，ward’s method [10]是聚类中一种常用方法，该方法对不相邻的特征(样本)的距离都要一一计算，计算量大，内存开销大，需要好的算法来处理这个问题[7]。尤其，当输入的语音序列比较大时，常规的凝聚型层次聚类 Hierarchical methods (HAC)方法就变得很慢。于是本文根据语音特征本身就具有相邻语音有较高的相似性，跨段语音差异更大的特点。提出基于 HAC 算法基础上相邻快速 HAC 算法解决时间长、内存开销大等问题，并应用到语音分割中来。

2. 层次聚类的原理及分类

2.1. 层次法(Hierarchical methods)

系统聚类也称层次聚类法。该算法先计算特征之间的距离，如欧式距离。在迭代过程中，每次查找到距离最近的点并合并到同一个类。然后，再计算新类与剩余类之间的距离，继续将距离最近的类合并为一个大类。不停的合并，直到合并到一个类或满足某个阀值。其中类与类的距离的计算方法有：最短距离法，最长距离法，中间距离法，类平均法等。比如最短距离法，将类与类的距离定义为类与类之间样本的最短距离[11]。

根据层次分解的顺序层次聚类算法可分为：自下向上和自上向下两种方法，即凝聚的层次聚类算法和分裂的层次聚类算法，也可以理解为自下而上法(bottom-up)和自上而下法(top-down)。

自下而上法的凝聚型层次聚类初始化时每个个体(object)都是一个类，然后根据 linkage 寻找同类，最后形成一个“类”[12]。

而分裂型层次聚类采用自上而下法，就是反过来，初始化时所有个体都属于一个“类”，然后根据 linkage 排除相似度不高的类，最后每个个体都成为一个“类”[11]。

2.2. 层次聚类的流程

凝聚型层次聚类的策略是先将每个对象作为一个簇，然后合并这些原子簇为越来越大的簇，直到所有对象都在一个簇中，或者某个终结条件被满足。绝大多数层次聚类属于凝聚型层次聚类，它们只是在簇间相似度的定义上有所不同。这里给出采用最小距离的凝聚层次聚类算法流程[13] [14]如下：

算法 1 HAC 算法
1) 将每个对象看作一个独立的类，计算两两之间的距离，并找出最小距离；
2) 将距离最小的两个类合并成一个新类；
3) 重新计算新类与所有类之间的距离；
4) 重复(2)、(3)，直到所有类最后合并成一类。

从上面算法 1 可以看出凝聚的层次聚类并没有类似基本 K 均值的全局目标函数，没有局部极小问题或是很难选择初始点的问题。合并的操作往往是最终的，一旦合并两个簇之后就不会撤销。当然其计算存储的代价是昂贵的。

2.3. 层次聚类的优缺点

层次聚类算法的优点有：距离和规则的相似度容易定义，限制少；不需预设聚类数 k；可以发现类的层次关系；可以聚类成其它形状。缺点有：计算复杂度太高，当特征(样本)数量很多就需要很多时间；奇异值也能产生很大影响；算法很可能聚类成链状。

这个方法其实效率比较低，特别是算 cluster 的 ESS 值还要先求均值点，然后算距离的平方再求和，不过有一个快速的计算方法叫 Lance-Williams Algorithm [15] (快速 HAC)可以大大简化 ward method 的计算。该算法可以不用 ESS 的公式计算 ESS，直接套用下面的公(1)。

$$D_{k,i+j} = \frac{1}{2\left(D_{ki} + D_{kj} - \left|D_{ki} - D_{kj}\right|\right)} \tag{1}$$

其中，初始的 ESS 由两点之间的距离决定，也就是说完全不需要算 ESS 了。但计算量和内存开销也大。因此，根据语音特征分割所具有的特点，进一步改进 HAC 算法有效解决语音特征分割面临的时间和空间的问题。

3. 改进 HAC 算法

从以上分析可以看出，原来的 HAC 或快速 HAC 算法都先把所有样本当成初始类，计算出样本之间的距离，并存储在矩阵中形成一个上三角矩阵[16]，从距离矩阵中找到最小的，合并成一个类，更新全部其他点到该新类的距离矩阵，再迭代上述步骤直到只剩一个类。整个过程的计算存储代价是昂贵的，耗时也严重，因语音特征中可能一长段语音特征对应同一个 phoneme，段内的语音特征相似度高，段外语音特征相似度不高的特点，只需关注相邻两两语音的相似，即左右特征的距离，不用关注不相邻样本间不相邻特征之间的距离来改进 HAC 算法并应用到语音分割中[17] [18]。

3.1. 算法设计

根据语音分割的特殊性，不用矩阵来存储距离，而是改用字典双链表来存储相邻特征距离，其伪代码如下：

$$Node = Node(i,\ j,\ dist) \tag{2}$$

其中，i，j 为相邻两两语音特征向量下标，$dist$ 其相邻语音特征的欧式距离[19]。欧式距离计算如下：

$$dist(X,Y) = \sqrt{\sum_{i}^{n}(x_i - y_i)^2} \tag{3}$$

其中，X, Y 为 n 维的语音特征(样本)，如 39 维的 MFCC。

同样，该算法初始时，每个特征是独立的 phoneme，在距离双链表中找到最小距离，和相邻语音特征下标，再根据下标来更新合并后的新数据点。

$$newdistance[clusterCount] = \frac{newdistance[front] + newdistance[back]}{2} \tag{4}$$

其中，$front$，$back$ 为相邻的两个特征下标，$newdistance[clusterCount]$ 为用平均法把 front 和 back 特征合并为新特征，就由合并的两个相邻特征的平均值来取代。

3.2. 算法流程

该算法中，凡是两个相邻的特征都计算它们的欧式距离[18]，相并逐一求出输入语音特征的相邻欧式距离，然后存表双链表中。先从相邻距离双链表中查找出距离最小的两个特征并合并成一个段，这个段就是那两个特征的平均值，接着再求新段与相邻特征的距离。层次聚类算法对语音特征分割结果示意图[19]如图 1 所示。

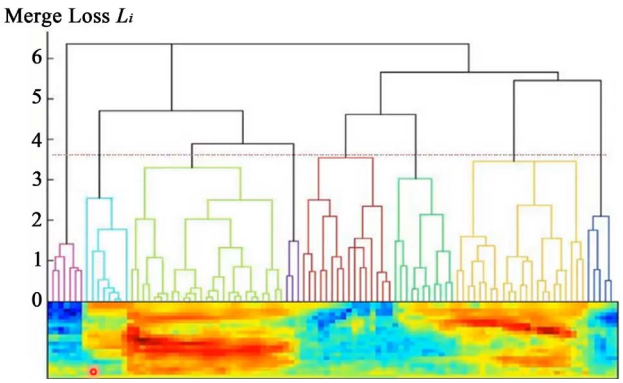

Figure 1. Schematic diagram of speech feature segmentation by hierarchical clustering algorithm
图 1. 层次聚类算法对语音特征分割结果示意图

对如图 1，横轴是时间，时间轴上为语音特征信号，如 39 维的 MFFC 序列。Merge Loss Li 为与合并前的那两个特征相差多少，最后长一整个树。当这棵树长好后，可以在这棵树上切一刀，就可以切出一个个段来。从虚线处切，虚线下一条线对应的每个子树就对应一个 segment，这样就得到 8 个 segment。如果觉得切的太细，可往上移再切。就可做任意精细度的切割法[7] [20] [21] [22]。

对快速 HAC 算法改进的相邻 HAC 算法流程如下所示：

算法 2 层次聚类 HAC 算法

1) 初始化聚类初始化相邻距离节点双链表；
2) 找出最小的相邻特征距离 minnode 及下标 front 和 back；
3) 合并相邻的两个类，并用平均法表示相邻两个特征并存入 newdistance[clusterCount]；
4) 更新找到最小类节点与相邻节点的距离并对应创建相邻新节点；
5) 删除两个相似子类及节点并更新相邻节点双链表；
6)只剩一个类或满足某个阀值则结束，如果不满足则迭代至 2)步。

其改进的相邻 HAC 算法伪代码如下：

1：clusterMap[k] = 1　# 存放 cluster 的情况，形如'1'：4 表示 cluster1 里面有 4 个元素(样本)

2：single_obj。append(i，i+1，dist)　#存放 cluster 之间的距离，形如'1，2'：3。22 表示 cluster1 与 cluster2 之间的距离为 3。22

#查找距离最短的两个 cluster，并迭代合并

3：while True：

4：minnode，　min = link。min() #查找距离最短的两个 cluster

5：front，　back = minnode。getName()　#取得相邻特征的下标

6：clusterMap[clusterCount]= ni + nj　　#合并相似段

#更新最小相邻距离 cluster 到新 cluster 的距离

7：newdistance[clusterCount] = (newdistance[front] + newdistance[back]) /2

#更新最小距离节点与相邻节点间的距离，并更新

8：Update_dist(minnode，left，right)

9：Update_link(minnode，left，right)

#删除最小节点和两个 cluster

10：del minnode，clusterMap[front，back]

#只剩一个子类或满足某个阀值

11：if len(clusterMap) == 2：

12：break　# 合并到只剩一个集合为止，然后退出

其中，伪代码第行和第行的两个函数 update_dist()和 update_link()分别为更新计算查找到距离最短的两个 cluster 并用平均法表示为一个新节点后与相邻类间的距离，以及重新更新查找到距离最短的两个 cluster 与相邻 cluster 的链表。

4. 实验对比与分析

4.1. 实验平台

本实验采用 python 3.6，硬件配置：CPU Intel(R) Xeon(R) E3-1231 v3 @ 3.4 GHz，16 G 内存，NVIDIA GeForce GTX 1060 3 GB；软件配置：pycharm64。

4.2. 实验结果

使用 62 Kb、199 Kb 和 1367 Kb 的 39 维的 MFCC 语音特征数据文件分别用快速 HAC 和改进的相邻 HAC 算法进行实验，得到对比结果，如表 1 所示。

Table 1. Fast HAC (GPU acceleration) and the performance comparison table of the improved adjacent HAC algorithm
表 1. 标准试验系统结果数据

算法名称	文件大小(KB)	理论时间复杂度(T(n))	实际运行时间	理论空间复杂度(S(n))
快速 HAC (GPU 加速)	62	O(n2)	20.53 s	O(n2)
	199		395.328 s	
	1367		35.62 h	
改进的相邻 HAC	62	O(n)	0.25 s	O(n)
	199		1.82 s	
	1367		81.49 s	

5. 结束语

HAC 层次聚类为无监督分类的重要工具，其默认分类算法中合并成本高，需要较大的计算力、较大的内存空间。如果总共有 n 个组，就必须把每个组合并都要算一遍，也要 $n \times (n-1)/2$，当特征(样本)数量增多时比较耗时，消耗内存也线性地增长。因此，为解决语音特征分割问题，提出相邻语音特征分割，只需关注相邻特征(样本)语音的合并问题，就无需要计算并更新所有特征或类间的距离，进而赢得了时间和空间，最后通过实验对比分析验证了算法的有效性。

参考文献

[1] 周涛, 袁飞, 庄旭. 最简数据挖掘[M]. 北京: 电子工业出版社, 2020.

[2] 邹臣嵩, 段桂芹. 基于改进 K-medoids 的聚类质量评价指标研究[J]. 计算机系统应用, 2019, 28(6): 235-242.

[3] Xie, W.-B., Lee, Y.-L., *et al*. (2020) Hierarchical Clustering Supported by Reciprocal Nearest Neighbors. *Information Sciences*, **527**, 279-292. https://doi.org/10.1016/j.ins.2020.04.016

[4] Lee, L.-S., Lee, H.-Y. and Chan, C.-A. (2015) Spoken Content Retrieval—Beyond Cascading Speech Recognition with Text Retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, **23**, 1389-1420.

[5] Qiao, Y., Shimomura, N. and Minematsu, N. (2008) Unsupervised Optimal Phoneme Segmentation: Objectives, Algorithm and Comparisons. *IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, 31 March-4 April 2008, 3989-3992. https://doi.org/10.1109/ICASSP.2008.4518528

[6] Wang, H.P., *et al*. (2015) Acoustic Segment Modeling with Spectral Clustering Methods. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **23**, 264-277. https://doi.org/10.1109/TASLP.2014.2387382

[7] Yang, S.-W., Liu, A.T. and Lee, H.-Y. (2019) Understanding Self-Attention of Self-Supervised Audio Transformers. *Computer Science*, **8**, 15-19.

[8] Jain, A.K. (2010) Data Clustering: 50 Years beyond k-Means. *Pattern Recognition Letters*, **31**, 651-666. https://doi.org/10.1016/j.patrec.2009.09.011

[9] Pratap, R., Deshmukh, A., Nair, P. and Dutt, T. (2018) A Faster Sampling Algorithm for Spherical k-Means. *Proceedings of Machine Learning Research*, Vol. 95, 343-358.

[10] HAC with Minimum SSE Criterion. https://hlab.stanford.edu/brian/error_sum_of_squares.html

[11] 李琳山. 数位语音处理概念. 2019. http://speech.ee.ntu.edu.tw/courses.html

[12] Novotney, S., Schwartz, R. and Ma, J. (2009) Unsupervised Acoustic and Language Model Training with Small Amounts of Labelled Data. *IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, 19-24 April 2009, 4297-4300. https://doi.org/10.1109/ICASSP.2009.4960579

[13] 吴信东，库玛尔. 数据挖掘十大算法[M]. 北京：清华大学出版社, 2014.

[14] 李勃昊，张连海，郑永军. 基于声学分段模型的无监督语音样例检测[J]. 数据采集与处理, 2016, 18(12): 41-44.

[15] Yarmish, G., Listowsky, P. and Dexter, S. (2017) Distributed Lance-William Clustering Algorithm. https://arxiv.org/ftp/arxiv/papers/1709/1709.06816.pdf

[16] Bhati, S., Nayak, S., Sri Rama Murty, K. and Dehak, N. (2019) Unsupervised Acoustic Segmentation and Clustering Using Siamese Network Embeddings. *INTERSPEECH* 2019, Graz, 15-19 September 2019, 2668-2672.

[17] Jansen, A. and Van Durme, B. (2011) Efficient Spoken Term Discovery Using Randomized Algorithms. *IEEE Automatic Speech Recognition and Understanding* (*ASRU*), Hawaii, 11-15 December 2011, 401-406. https://doi.org/10.1109/ASRU.2011.6163965

[18] Badino, L., Canevari, C., Fadiga, L. and Metta, G. (2014) An Autoencoder Based Approach to Unsupervised Learning of Subword Units. *Acoustics*, *Speech and Signal Processing* (*ICASSP*), Florence, 4-9 May 2014, 7634-7638. https://doi.org/10.1109/ICASSP.2014.6855085

[19] 詹竣安. 以口語查詢之非督導式口語詞彙偵測[D]: [博士学位论文]. 台北：台湾大学, 2012.

[20] Mary, L. and Deekshitha, G. (2018) Searching Speech Databases: Features, Techniques and Evaluation Measures. Springer, Berlin. https://doi.org/10.1007/978-3-319-97761-4

[21] Nazari, Z. and Kang, D. (2018) A New Hierarchical Clustering Algorithm with Intersection Points. *IEEE Uttar Pradesh Section International Conference on Electrical*, *Electronics and Computer Engineering*, Gorakhpur, 2-4 November 2018, 315-319. https://doi.org/10.1109/UPCON.2018.8596795

[22] Chung, C.T. and Lee, L.S. (2018) Unsupervised Discovery of Structured Acoustic Tokens with Applications to Spoken Term Detection. *IEEE/ACM Transactions on Audio*, *Speech*, *and Language Processing*, **26**, 394-405.