doi: 10.3969/j.issn.1003-4271.2014.06.14

一种自适应 AP 算法的 matlab 实现

向培素

(西南民族大学电气信息工程学院, 四川 成都 610041)

摘 要: AP 算法是 FeyBJ.等人提出的一种聚类算法. 与传统的 K 均值聚类算法相比, AP 算法不需要选择初始的聚类中心点, 因此, 聚类结果更客观. 但 AP 算法中相似度矩阵对角线上的偏向值需要人为设定, 而这个值会影响到聚类数目; 另外, 当 AP 算法发生震荡时, 算法无法自动退出震荡. 为解决 AP 算法中的振荡问题及相似度矩阵对角线上元素值的确定问题, 王开军等人提出了自适应 AP 算法, 逐步改变偏向值 p, 得到不同的聚类结果, 再根据聚类结果的 Silhouette 指标, 找出最好的 Silhouette 指标对应的偏向值及聚类结果. 当震荡发生时, 逐步增加阻尼因子 λ 值, 直到算法退出震荡. 使用 MATLAB 实现了自适应 AP 算法和 Silhouette 评价指标, 为后续的研究工作打下基础.

关键词: 自适应 AP 算法; Silhouette 指标; 聚类算法; Matlab

中图分类号: TP301.6 **文献标识码**: A **文章编号**: 1003-4271(2014)06-0877-06

1 引言

在自然科学和社会科学中, 存在着大量的分类问题. 聚类和分类的不同之处在于, 聚类对象的类别是未知的, 而分类对象的类别是已知的. 这个特点非常适用于当前大数据背景下的数据挖掘. 随着各个领域数据的增多, 数据聚类分析成为一个非常重要的分析工具.

AP(Affinity propagation clustering)算法[1]是在贝叶斯网络[2]的基础上, 由对数域中的和积算法[3]——最大和算法推导出来的. 这个算法通过在数据点之间传递消息, 来获得使各个数据点与所属类的类代表点的相似度之和取最大值的聚类结果.

正如王开军等人指出的, AP 算法有两个缺陷, 一个是在计算相似度矩阵时, 其对角线上的取值只能人手工设定, 这就很难设定为最优值, 而这个取值会影响到最后的聚类结果. 另一个缺陷是当振荡发生后, 就很难自己停下来. 为解决这两个问题, 提出了自适应 AP 算法[4-5].

鉴于文献[6]已无效, 无法获取算法源程序, 本文使用 matlab 重新实现了王开军等人提出的自适应 AP 算法, 为后期的研究工作打下了基础.

本文分为两个部分: 1)介绍了 AP 算法和自适应 AP 算法, 并用 matlab 实现了自适应 AP 算法; 2)介绍了 Silhouette 聚类评价指标, 并使用 matlab 在自适应 AP 算法中实现了该评价指标的计算.

2 AP 算法和自适应 AP 算法

2.1 AP 算法

在AP算法中, 对N个数据点的聚类, 可以看做是寻找使各个数据点与其所属类代表点相似度之和最大的那种聚类方式. 用公式可以表示为:

收稿日期: 2014-10-08

作者简介: 向培素(1974-), 女, 汉族, 湖北人, 副教授, 主要研究方向: 计算机应用.

基金项目: 2012 年度西南民族大学中央高校基本科研业务费专项项目(12NZYQN05).

$$\arg\max S(c) = \arg\max(\sum_{i=1}^{N} s(i,c_i) + \sum_{k=1}^{N} \delta_k(c)),$$

其中 i 是第 i 个数据点，c_i 指第 i 个数据点的类代表点，$s(i,c_i)$ 指第 i 个数据点与其类代表点之间的欧几里得距离的负数，$\delta_k(c)$ 是一个“惩罚指标”，当某个类的类代表点 k 同时又是别的聚类中的成员时，这种聚类是不合理的(类代表点 k 应该在以自己为类代表点的聚类中)，应该被排除，故此时 $\delta_k(c) = -\infty$，其余情况，$\delta_k(c) = 0$.

上面等式的右边，正好适用于最大和算法. 由于最大和算法是对数域的和积算法. 因此，可以使用因子图来表示上面的等式. 由此推导出了下面的递推迭代公式:

$$r(i,k) = s(i,k) - \max_{k' s.t. k' \neq k} \{a(i,k') + s(i,k')\},$$

$$a(i,k) = \min\{0, r(k,k) + \sum_{i' s.t. i' \notin \{i,k\}} \max\{0, r(i',k)\}\},$$

$$a(k,k) = \sum_{i' s.t. i' \neq k} \max\{0, r(i',k)\},$$

其中 $a(i,k)$ 的初值为 0. 算法结束条件可以是迭代指定次数、或者传递的 a 值，r 值的变化很小、或者是类代表点的选择不再改变. 为避免振荡，在迭代时，引入了 0~1 之间的阻尼因子 λ:

$$r = (1-\lambda) * r + \lambda * rold, \quad a = (1-\lambda) * a + \lambda * aold$$

其中 rold 指上次迭代计算出的 r 值, aold 指上次迭代计算出的 a 值. 从算法结束条件及为避免振荡采取的措施来看，一旦发生振荡，算法是无法自行结束的.

2.2 自适应 AP 算法

自适应 AP 算法包括了三部分内容: 1.自适应阻尼: 如果算法发生振荡，则逐步增加阻尼因子 λ 的值以消除振荡. 2.自适应逃离: 如果在 λ >=0.85 时，算法仍然振荡，则降低相似度矩阵对角线上元素的值(p 值)，以退出振荡，并使用新的 p 值重新进行迭代计算. 3.自适应扫描: 逐步减少 p 值会导致不同的聚类数目，以具有最优 Silhouette 评价指标的聚类结果为最后的聚类结果.

自适应阻尼算法中，采用“非振荡特征”的出现次数来判断是否发生振荡. “非振荡特征”是指迭代产生的新的聚类数目等于或者小于本次迭代前的聚类数目. 也就是说，迭代产生的聚类数目不变或者减少，则认为没有发生振荡.

自适应逃离算法中, p 值减少的步幅值由实验确定为: $0.01 * pm/(0.1 * \sqrt{K+50})$

自适应扫描算法，是将相似度矩阵 s 对角线上元素的值设为矩阵 s 所有其它元素的均值的一半，然后进行迭代运算，当算法收敛后，记录下来. 再将 p 值减少，进行迭代运算，待收敛后，再记录下来...持续这个过程，直到聚类数 K=2，或者整个迭代次数超过 50000 次. 然后将所有记录下来的聚类结果进行比较，选取 Silhouette 评价指标最优的那个作为最后的聚类结果.

算法的 matlab 实现:

```
matrix=data;                          %数据矩阵的名称,使用 matlab 软件的 Import Data 功能把 UCI 数据
                                      %breast cancer wisconsin 加载入 matlab，形成的矩阵名称叫 data.
N=size(matrix,1);                     %数据集中数据的个数
LEI=2;                                %数据集的实际类别数
attribute=size(matrix,2);             %数据集中数据的维数,列是 1，行是 2，故指每行的维数
s=zeros(N,N);                         %s 为相似度矩阵，初值为 0
for k=1:N;
    for j=1:N;
      for i=1:attribute;
        if k==j
            s(k,j)=2;
```

```
        else s(k,j)=s(k,j)+(matrix(k,i)-matrix(j,i))*(matrix(k,i)-matrix(j,i));     %求各数据点的相似度,
                                                                                    %用欧氏距离表示.
        end
      end
        s(k,j)=-s(k,j);
      end
end
pm=mean(s);                 %mean 函数将 s 的列看做向量，返回每列的均值，构成一个行向量
pm=mean(pm,2);              %相似度矩阵的均值
   for k=1:N
       s(k,k)=pm/2;         %相似度矩阵对角线上的值设为均值
   end
   p=pm/2;

lam=0.5-0.05;               %因为迭代前阻尼因子会+0.05,因此这里-0.05，保证迭代时阻尼因子从 0.5 开始.

A1=zeros(N,N);
R1=zeros(N,N);
Kb1=zeros(40);              %Kb 是自适应 AP 算法中的“移动监视窗”，用于记录出现非振荡特征的次数,
                            %取 40 个是因为 iter=1:40

w=0;                        %迭代次数
K1=1;                       %K1 是最后的聚类个数
I1=zeros(K1);               %每个元素是一个类代表点的下标
Kall=[];                    %存放不同 p 值对应的聚类结果中的聚类数
idxall=[];                  %存放不同 p 值对应的聚类结果中每个数据点所属的类代表点下标.
Iall=cell(N,1);             %存放不同 p 值对应的聚类结果中类代表点的下标.
ka=0;
Kold1=K1;
Kold10=3;                   %Kold10 指迭代 40 次前，聚类个数，与迭代 40 次中的每次迭代产生的 Kold1
                            %相区别．使初始状态 Kold10>=K1
Iold10=I1;                  %使初始状态 Iold10>=I1
b=-1;
    k1=0;                   %聚类结果不变的次数
while(Kold10>2&Kold10<sqrt(N))
    k1=0;                   %聚类结果不变的次数
while(k1<=10)               %若聚类结果不变的次数小于 10，则继续迭代
k0=1;                       %若未发生震荡，则 k0 为 1，否则 k0 为 0

if(sum(Kb1)<2/3*length(Kb1))      %若非振荡特征出现的次数少于 2/3Kb 窗口的宽度，则认为发生振荡
     k1=0;                  %k1 记录聚类结果不变的次数，发生震荡，则得到的聚类结果无效，故清零
     k0=0;                  %k0=0 表示发生震荡
     if(lam<=0.85)
    lam=lam+0.05;
     else
         b=b+1;
         p=p-b*0.01*pm/(0.1*sqrt(K1+50));      %若 lam 已经大于 0.85 仍发生振荡，则减少 p 值.
         for k=1:N
```

```
                s(k,k)=p;
            end
        end          %若发生振荡，则将 lam 值增加或是减少 p 值，再重新迭代 40 次，否则，记录下聚类结果未改变的次数等于 10，就记录下聚类结果，改变 p 值，进行下一次聚类结果的迭代.
    elseif k0==1;      %如果未发生震荡，则 k0=1
        k1=k1+1;
    end
  for iter=1:40
     此处为 AP 核心算法.
  w=w+1;                                          %w 是迭代次数，当 w>50000 次时，退出循环.
      E1=R1+A1;
     I1=find(diag(E1)>0);K1=length(I1);
     [tmp c]=max(s(:,I1),[],2);c(I1)=1:K1;idx1=I1(c);      %I 为聚类中心点下标
   end
   if (Kold10~=K1|Iold10~=I1)
     k1=0;
   end
  Kold10=K1;                                      %记录迭代 40 次后，得到的结果
  Iold10=I1;
 end
 if(w>50000)break;
 end
 b=b+1;                                           %自适应 AP 算法中定义的 b
 p=p-b*0.01*pm/(0.1*sqrt(K1+50));                 %自适应 AP 算法中的 p 值递减公式
            for k=1:N
                s(k,k)=p;
            end
 ka=ka+1;
 Kall(ka)=K1;
 Iall{ka}=I1;
 idxall{ka}=idx1;
 if(K1<=2) break;
 end
 end
```

3 Silhouette 聚类评价指标

一个数据点的 silhouette 指标计算公式为: $S_{il}(t)=\dfrac{b(t)-a(t)}{\max\{a(t),b(t)\}}$，其中 $a(t)$ 为某个聚类中点 t 与聚类中所有其它点的平均距离. $b(t)$ 为某个聚类中的数据点 t 与别的聚类中的所有数据点的平均距离的最小值.

这些距离都可以通过相似度矩阵求得.

自适应 AP 算法中，该指标的 matlab 代码如下:

```
idxa=zeros(N,N);
idxb=zeros(N,N);
d=ones(N,size(I1,1));
sil=zeros(N,1);
Sav=zeros(ka,1);
for ix=1:ka
idx=idxall{ix};          %每次循环取出一个 p 值对应的聚类结果中各数据点所属类代表点的下标.
```

```
I=Iall{ix};                          %取出一个 p 值对应的聚类结果中类代表点的下标
for t=1:N
        for k=1:N
        if k~=t
          if idx(k)==idx(t)
              idxa(t,k)=1;                    %将相同类的数据点标记为 1
          end
        end
        end
        a(t)=mean(idxa(t,:).*-s(t,:));        %t 点与自己聚类中其它点的距离均值.

        for m=1:size(I,1)%对 m 个聚类一次循环
            if idx(t)~=I(m)
            for k=1:N
                if idx(k)==I(m)
                    idxb(t,k)=1;              %将聚类 I(m)中的数据点标注为 1
                end
            end
              d(t,m)=mean(idxb(t,:).*-s(t,:));   %t 点与 m 类中各数据点的距离
              idxb(t,:)=0;
            else d(t,m)=10000000000;
            end
        end
        b(t)=min(d(t,:));                     %点 t 与别的聚类中各数据点平均距离的最小值
        sil(t)=(b(t)-a(t))/max(a(t),b(t));
end
Sav(ix)=mean(sil(:,1));
end
```

4　结论

针对 AP 算法偏向值 p 无法自动设定，及算法发生震荡，无法自动退出的问题，王开军等人提出了自适应 AP 算法，本文中使用 matlab 实现了自适应 AP 算法和 Silhouette 聚类评价指标，该程序可以作为 matlab 的工具箱应用于图像检索、图像分割、图像识别、设施选址、文本挖掘等领域，也可以作为聚类算法研究的基础：在此基础上进行进一步的聚类算法优化，或者不同聚类算法的特征比对，或者新的聚类算法的应该领域，应用方法的研究.

参考文献

[1] FreyBJ, DUeCkD. Clustering by Passing messages between data points[J]. Science, 2007, 315: 972-976.

[2] JUDEA PEARL. Fusion, Propagation, and Structuring in Belief Networks [J]. Artificial Intelligence , 1986, 29: 241-288.

[3] FRANK R, KSCHISCHANG,BRENDAN J, et al. Factor Graphs and the Sum-Product Algorithm[J]. Trans Inform Theory , 2001, 47(2): 498-519.

[4] KAIJUN WANG, JUNYING ZHANG, DAN LI, et al. Adaptive Affinity Propagation Clustering [J]. Acta Automatica Sinica, 2007, 33(12): 1242-1246.

[5] 王开军，张军英，李丹，等. 自适应仿射传播聚类[J]. 自动化学报, 2007, 33(12): 1242-1246.

[6] WANG K J. Supplement of adaptive affinity propagation clustering[EB/OL]. (2007-12-11)[2014-03-19]http://www. mathworks. com/ matlabcentral/ fileexchange/loadAuthor.do?objectType=arthor&objectId=1095267.

[7] 向培素. 近邻半监督聚类算法的 MATLAB 实现[J]. 数字技术与应用, 2012, 08: 100-101.

[8] HALKIDI M, VAZIRGIANNIS M, BATISTAKIS Y. Quality scheme assessment in the clustering process [A]. Proc of 4th Eur Conf Principles and Practice of Knowledge Discovery in Databases[C]. 2000:165-276.

[9] 张连文，郭海鹏. 贝叶斯网引论[M]. 北京：科学出版社, 2006.

[10] 张殿祜，方绍辉，丁潇君. 熵——度量随机变量不确定性的一种尺度[J]. 系统工程与电子技术, 1997, 11: 1-8.

[11] 陈峰, 刘红, 徐文立. 递推信度传播算法——按良序的信度传播[J]. 自动化学报, 2010, 36(8): 1091-1098.
[12] 杨燕, 靳蕃. KAMEL Mohamed 聚类有效性评价综述[J]. 计算机应用研究, 2008, 25(6): 1630-1638.
[13] 向培素. 聚类算法综述[J]. 西南民族大学学报: 自然科学版, 2011(1): 112-114.
[14] 张惟皎, 刘春煌, 李芳玉. 聚类质量的评价方法[J]. 计算机工程, 2005, 31(20): 10-12.
[15] 孙吉贵, 刘杰, 赵连宇.聚类算法研究[J]. 软件学报, 2008, 19(1): 48-61.
[16] 周世兵, 徐振源, 唐旭清.基于近邻传播算法的最佳聚类数确定方法比较研究[J].计算机科学, 2011,38(2):225-228.

The MATLAB program designing of adaptive AP algorithm

XIANG Pei-su

(School of Electrical & Information Engineering, Southwest University for Nationalities,Chengdu 610041, P.R.C.)

Abstract: The AP algorithm is a kind of clustering algorithm proposed by FeyBJ.et al. Compared with the traditional k-means clustering algorithm, the AP algorithm does not need to select the initial exemplare. Therefore, the cluster results are more objective. The diagonal of the similarity matrix in AP algorithm is hard to determine, and the value will affect the clustering number. In addition, the AP algorithm oscillate algorithm cannot automatically exit. To solve the problem of oscillation of the AP algorithm and to determine the diagonal element value of the similarity matrix, WANG Kai-jun et al. proposed adaptive AP algorithm, changing p step by step, obtain the different clustering result, according to the clustering results's Silhouette index, find out the best clustering results. When oscillations occurs, AP algorithm increases the damping factor value step by step, until the oscillation stops. The paper proposed a MATLAB programming of adaptive AP and Silhouette Index. It provides a foundation work for further study.

Key words: adaptive AP; silhouette; clustering algorithm; Matlab