软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2016,27(11):2796−2813 [doi: 10.13328/j.cnki.jos.004921]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

迁移近邻传播聚类算法*

杭文龙, 蒋亦樟, 刘解放, 王士同

(江南大学 数字媒体学院,江苏 无锡 214122)

通讯作者: 杭文龙, E-mail: hwl881018@163.com

摘 要: 在目标域可利用数据匮乏的场景下,传统聚类算法的性能往往会下降.在该场景下,通过抽取源域中的有用知识用于指导目标域学习以得到更为合适的类别信息和聚类性能,是一种有效的学习策略.借此提出一种基于近邻传播的迁移聚类(transfer affinity propagation,简称 TAP)算法,在源域和目标域数据分布相似的情况下,通过引入迁移学习机制来改善近邻传播聚类(affinity propagation,简称 AP)算法在数据匮乏场景下的聚类性能.为保证迁移的有效性,TAP 在综合考虑源域和目标域的统计特性及几何特征的基础上改进 AP 算法中的消息传递机制使其具备迁移能力,从而达到辅助目标域学习的目的.此外,通过 TAP 对应的因子图,亦可说明 TAP 可以以类似 AP 的消息传递机制,在目标域数据匮乏的情况下进行高效的知识迁移,为最终所获得的聚类结果提供了保证.在模拟数据集和真实数据集上的仿真实验结果显示,所提出的算法较之经典 AP 算法在处理非充分数据聚类任务时具有更佳的性能.

关键词: 迁移学习;统计特征;几何结构;近邻传播;聚类方法;非充分数据

中图法分类号: TP181

中文引用格式: 杭文龙,蒋亦樟,刘解放,王士同.迁移近邻传播聚类算法.软件学报,2016,27(11):2796−2813. http://www.jos.org.cn/1000-9825/4921.htm

英文引用格式: Hang WL, Jiang YZ, Liu JF, Wang ST. Transfer affinity propagation clustering algorithm. Ruan Jian Xue Bao/ Journal of Software, 2016,27(11):2796−2813 (in Chinese). http://www.jos.org.cn/1000-9825/4921.htm

Transfer Affinity Propagation Clustering Algorithm

HANG Wen-Long, JIANG Yi-Zhang, LIU Jie-Fang, WANG Shi-Tong

(School of Digital Media, Jiangnan University, Wuxi 214122, China)

Abstract: The main limitation of most traditional clustering methods is that they cannot effectively deal with the insufficient datasets in target domain. It is desirable to develop new cluster algorithms which can leverage useful information in the source domain to guide the clustering performance in the target domain so that appropriate number of clusters and high quality clustering result can be obtained in this situation. In this paper, a clustering algorithm called transfer affinity propagation (TAP) is proposed for the insufficient dataset scenarios. The new algorithm improves the clustering performance when the distribution of source and target domains are similar. The basic idea of TAP is to modify the update rules about two message propagations, used in affinity propagation (AP), through leveraging statistical property and geometric structure together. With the corresponding factor graph, TAP indeed can be applied to cluster in AP-like transfer learning, i.e., TAP can abstract the knowledge of source domains through the two tricks to enhance the learning of target domain even if the data in the current scene are not adequate. Extensive experiments demonstrate that the proposed algorithm outperforms traditional algorithms in situations of insufficient data.

Key words: transfer learning; statistical property; geometric structure; affinity propagation (AP); cluster method; insufficient data

* 基金项目: 国家自然科学基金(61272210, 61202311, 61300151); 江苏省自然科学基金(BK2012552, BK20130155)

Foundation item: National Natural Science Foundation of China (61272210, 61202311, 61300151); Natural Science Foundation of Jiangsu Province (BK2012552, BK20130155)

收稿时间: 2014-10-29; 修改时间: 2015-03-18, 2015-05-14; 采用时间: 2015-09-19; jos 在线出版时间: 2015-11-27

CNKI 网络优先出版: 2015-11-26 16:06:06, http://www.cnki.net/kcms/detail/11.2560.TP.20151126.1606.001.html

近邻传播(affinity propagation,简称 AP)算法[1]因其独特的聚类原理,自 2007 年由 Frey 等人提出后,一直受到广大研究者的关注.根据 Frey 等人于文献[1]中的介绍可知,AP 算法的本质实际上是一种基于因子图[2]的信念传播和最大化算法.与其他经典聚类方法相比,AP 算法具有以下几点优势:(1) AP 聚类不需要指定 K(经典的 K-Means[3])或者是其他描述聚类个数(SOM[4]中的网络结构和规模)的参数;(2) 一个聚类中最具代表性的点在 AP 算法中叫做聚类代表点(exemplar),与其他算法中的聚类中心不同,聚类代表点是原始数据中确实存在的数据点,而非虚拟点;(3) 多次执行 AP 聚类算法,得到的结果是完全一样的,不需要进行随机选取初值步骤,即 AP 算法对初始化不敏感;(4) AP 聚类比其他方法的误差平方和都要低[5,6].由于其鲁棒性较强,大量基于 AP 的增强算法不断被提出来,如软约束 AP[7]、层次 AP[8]、半监督 AP[9]等.

AP 算法较为成功的特点是能够自动产生合理的聚类个数,在数据比较充足的情况下,AP 能够准确地找出聚类代表点,其最终所获得的聚类效果往往也较为理想.然而在实际应用中,由于某些生产过程数据的保密性较高,抑或高代价产业导致的低产量等,收集到的数据样本通常十分有限,从而造成数据匮乏的场景时常出现.在面对这种场景下的聚类任务时,由于 AP 算法得到聚类代表点往往对数据的几何分布较为敏感,这主要是由于 AP 旨在最大化各类别中数据点到其聚类代表点的能量之和,往往忽略了数据匮乏场景下的真实几何分布,难以满足除能量最小之外的任何要求,导致所获得的聚类代表点和分配矩阵不够准确.因此,在数据量严重匮乏的情况下,若继续使用 AP 算法的能量最小原则,而忽略相关领域的重要信息,则会影响最终的聚类效果.因此,如何使 AP 算法在面对数据匮乏场景时仍然能够具备较好的类别辨识能力和较高的聚类性能,是当前丞待解决的问题.

迁移学习框架[10−12]与人类的认知过程类似,能够有效利用已有知识用于指导新事物的学习过程,其被证明能够有效地解决在数据匮乏场景下的机器学习问题[13].近年来,迁移学习框架被大量应用于模式分类、回归建模及聚类等方面,其中具有代表性的工作有:

(1) 在模式分类领域上,Raina 等人在 2007 年将迁移学习的理论应用到了未标记数据的分类问题上[14];Xue 等人在文献[15]中提出了基于迁移学习理论的 TPLSA 算法并将其应用于文本分类;文献[16]中,Glorot 等人进一步将领域适应的迁移学习方法成功地应用到了大规模情感数据的分类问题;Dai 提出了基于 boosting 算法的迁移学习方法[17],用于解决源域有干扰数据情况下的分类问题;Tommasi 等人提出了一种多源自适应迁移学习策略[18],用于图像分类问题.

(2) 在回归建模领域上,Deng 等人分别提出了基于知识利用的迁移模糊系统以及增强版本,均用于解决在数据匮乏场景下的模糊回归建模问题[13,19,20].

(3) 在无监督聚类领域,目前的研究还较少.2012 年,Jiang 等人提出了一种基于谱方法的迁移聚类算法,解决了文本类数据的迁移聚类问题[21].

根据上述研究成果可以发现,当前的迁移学习策略旨在从源域数据中抽象出相关的有效知识用于指导目标域的学习,其主要的学习策略可总结为以下两点:(1) 保留统计特性,即最大化嵌入方差或最小化重构误差[13−19,21];(2) 保持几何结构,即,使目标域中类似的数据保持与源域中样本类似的表达方式[22−26].

尽管上述研究工作在不同的应用上均取得了令人满意的成果,但以往这些工作均在迁移过程中仅选用了一种学习模式而忽视了其他结构信息的作用.为了弥补 AP 算法在数据匮乏时聚类性能下降的缺陷,本文将迁移学习机制引入到 AP 算法中,从而得到具备知识迁移能力的迁移 AP 聚类算法.此外,在知识迁移过程中同时考虑数据的统计特性和几何结构,保证迁移的质量.本文所提出的迁移 AP 算法可看做是 AP 算法的一种泛化,它能够有效地在相似的领域或任务之间进行信息共享和迁移,同时又保持 AP 的经典形式.本文将重点介绍与分析两种迁移策略以及迁移聚类算法.本文将首先介绍 AP 算法和两种迁移策略,然后介绍集成这两种策略的基于近邻传播的迁移聚类方法 TAP(transfer affinity propagation).

1 近邻传播(AP)聚类算法

AP 算法以 N 个数据点之间的相似度矩阵 $\boldsymbol{S}$ 为初始输入,在算法开始时将每一个数据点都视为潜在的聚类代表点(exemplar),其旨在找出合适的聚类代表点,使得聚类能量达到最小.在 AP 算法中,能量通常使用距离进行

数值化表征,其常采用负欧氏距离[1],即最大化如下目标函数:

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

这里,$s(i,c_i)$表示第 i 个点到第 c_i 个潜在聚类代表点的距离.$\delta_k(c)$是对潜在代表点 c_k 的惩罚,若数据点 c_i 选择 k 作为其类代表点,即 $c_i=k$,那么数据点 k 必须选择自身作为类代表点,即 $c_k=k$;否则,数据点 k 不能成为聚类代表点.

$$\delta_k(c)=\begin{cases}-\infty, & \text{if } c_k\neq k \text{ but } \exists i: c_i=k\\ 0, & \text{otherwise}\end{cases} \tag{2}$$

如图 1 所示,AP 算法通过基于因子图的信念传播和最大化来达到最优化.

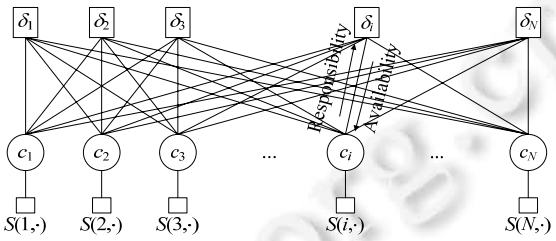


Fig.1 Factor graph of the AP method

图 1 AP 算法因子图

该算法将两点之间的欧式距离负的值看作它们之间的相互吸引度,即,若点 k 对较近的点 i 吸引度比较大,同样,点 i 认同点 k 为其聚类中心的归属度也较大.若数据点 k 对其他数据点的吸引力之和较大,则其成为聚类中心的可能性也越大;反之,处于聚类边缘处的数据点对其他数据点的吸引力和较小,则其成为聚类中心的可能性也越小.在 AP 算法迭代过程中传递两类消息:一个是候选代表点 k 从每个数据点 i 获取的正向信息 $r(i,k)$,用来描述数据点 k 合适作为数据点 i 的类代表点的程度;另一个是数据点 i 从候选类代表点 k 获取的逆向信息 $a(i,k)$,用来描述数据点 i 选择数据点 k 作为其类代表点的适合程度(如图 1 所示).两类消息初始化都为 0,并按如下算式各自迭代:

$$\begin{cases}r(i,k)\leftarrow s(i,k)-\max\limits_{j\neq k}[s(i,j)+a(i,j)]\\ a(i,k)\leftarrow \min\left[0,r(k,k)+\sum\limits_{i'\in\{i,k\}}\max[0,r(i',k)]\right], & k\neq i\\ a(k,k)\leftarrow \sum\limits_{i'\neq k}\max[0,r(i',k)]\end{cases} \tag{3}$$

当算法收敛或达到一定迭代次数时,各个数据点的标签分配向量 $c=[c_1,\ldots,c_N]$通过公式(4)计算得到:

$$c_i=\arg\max_j[a(i,j)+r(i,j)] \tag{4}$$

可以看出,在数据量充足时,AP 算法可以在几乎无需外部干预的情况下高效地找出最佳聚类代表点和分配矩阵;但在数据量匮乏的情况下,若没有任何外加的加强学习条件,仍然使用原始的聚类法则,则极易使聚类中心点发生较大的偏差,导致聚类失效.本文将针对此不足,给出一种全新的迁移聚类方法.

2 迁移近邻传播(TAP)聚类算法

针对上节所述的 AP 聚类方法在数据匮乏时的不足,本文基于数据的统计特征和几何结构引入两大技术对当前方法进行有效的改进.我们提出一种新的适用于 AP 算法的迁移学习框架,该框架将充分利用源域数据的统计特征(分布匹配迁移策略)以及源域数据和目标域数据之间的几何特征(实例保留迁移策略)来提高迁移聚类效果,保证整个迁移学习的质量,增强 AP 算法在面对数据匮乏场景下的聚类效果,具体方法见下一节.

2.1 领域分布近似策略

根据迁移学习理论可知,如果领域之间的分布相似度越高,即目标域的分布越接近源域,那么由源域抽象出的知识指导目标域数据学习时,理论上正确率也应该得到提升.利用聚类任务之间的数据分布相似性原则,我们从源域中随机抽出部分数据,源域与目标域的分布越接近,这部分数据属于源域各自类代表点的能量与此部分数据属于目标域中类代表点的能量就越接近,如图 2 所示.

Fig.2 Demonstration of preserving the statistical property

图 2 保持统计特性示例图

用 $c=[c_1,\ldots,c_N]$表示分配矩阵.目标域所有数据点和对应的类代表点的能量之和用 S_1 表示,选自源域的数据点 j 隶属于源域真实代表点 c_i' 的能量与其隶属于目标域潜在类代表点 c_i 的能量的差异平方和用 S_2 表示,分别表示为

$$S_1=\sum_{i=1}^{N}s(i,c_i) \tag{5-1}$$

$$S_2=\sum_{j=1}^{N'}[s(j,c_i)-s(j,c_i')]^2 \tag{5-2}$$

其中,c_i 表示目标域聚类潜在代表点,c_i' 表示源域的聚类代表点,N'表示选自源域数据点的个数.定义一个函数矩阵 $[S_{ic_i}]_{N\times N}$,表示目标域中所有的数据与潜在类代表点的能量关系,其表示如下:

$$S_{ic_i}=s(i,c_i)+\lambda_1\cdot[s(j,c_i)-s(j,c_i')]^2 \tag{6}$$

其中,λ_1 为待定参数,用于惩罚源域和目标域的分布差异.由公式(5-1)、公式(5-2)、公式(6),我们可得到 $S_1+\lambda_1\cdot S_2=\sum_{i=1}^{N}S_{ic_i}$.这里,$\lambda_1$ 的取值范围为[0.1,1],间隔 0.1.

2.2 领域聚类代表点几何迁移策略

正如文献[12]中所述,迁移学习需要处理两种情况:(1) 用于学习的训练样本与新的测试样本不满足独立同分布的条件;(2) 没有足够可利用的训练样本.可以看出,在可利用数据匮乏的情况下,分布属性有时并不能保证迁移的有效性.因此,我们利用源域的类代表点与目标域的类代表点的几何特征来保障迁移的可行性.近邻数据在一定程度上表达了数据的流形几何信息,因此,利用源域聚类代表点的信息辅助目标域聚类代表点的选择可以使得在目标域数据匮乏时聚类更加准确.如图 3 所示,黑色和红色数据点为源域数据集,其中,红色为其类代表点;绿色和黄色数据点为目标域数据集,黄色为其类代表点.很明显,源域数据中的代表点及其近邻可以用来帮助目标域的学习.此外,该迁移策略还可以加快 AP 算法的收敛速度,我们将在第 2.3 节中介绍.

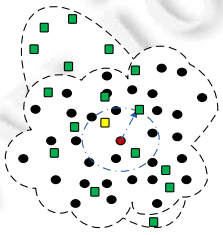


Fig.3 Demonstration of preserving the property of geometric structure

图 3 几何特征保留示例图

根据图 3,为使 AP 算法在迁移学习过程中进一步利用源域数据与目标域数据之间的几何结构,我们先进行如下的定义.

定义 1(对象 x 的ε近邻). 数据点对象 x 的ε近邻表示为 $neb(x)$,定义如下:

$$neb(x)=\{y\in D|dist(x,y)\leqslant\varepsilon\} \tag{7}$$

这里,$dist(y,x)$为距离度量函数,表示 x,y 之间的距离;ε为阈值.本文中的 $dist(y,x)$函数将选用欧式距离.

源域聚类代表点对目标域数据的聚类具有指导作用,由于源域和目标域具有相似的分布特性和几何分布属性,源域数据聚类代表点近邻内的目标域数据具有更大的可能性成为聚类代表点.因此,我们对目标域该近邻内数据中的潜在类代表点的做如下惩罚:

$$\Delta_k(\boldsymbol{c})=\begin{cases}-\infty, & \text{if } c_k\neq k \text{ but } \exists c_i=k,\\ \lambda_2\cdot I, & \text{if } c_k=k \text{ and } (\exists j,c_k\in neb(c'_j))\\ 0, & \text{otherwise}\end{cases} \tag{8}$$

这里,I 定义如下:

$$I=\frac{1}{N}\sum_{i=1}^{N}s(i,c_k) \tag{9}$$

c_k 为潜在类代表点,λ_2 为惩罚系数,具体取值在实验部分给出.对比上式与公式(2)可以发现,$\Delta_k(\boldsymbol{c})$类似于 AP 中的 $\delta_k(\boldsymbol{c})$,其表示为对潜在代表点 $\boldsymbol{c}$ 的惩罚.此外,值得说明的是,$\Delta_k(\boldsymbol{c})$对落在源域聚类代表点近邻的目标域潜在聚类代表点做出$\lambda_2\cdot I$ 的惩罚,增大了此类潜在类代表点成为最终类代表点的可能性.

综上,所提出的 TAP 算法的目标函数可定义如下:

$$\max S(c)=S_1+\lambda_1\cdot S_2+\Delta_k(c)=\sum_{i=1}^{N}S_{ic_i}+\Delta_k(c)=\sum_{i=1}^{N}s(i,c_i)+\lambda_1\cdot\sum_{j=1}^{N'}[s(j,c_i)-s(j,c'_i)]^2+\Delta_k(c) \tag{10}$$

对比公式(10)中的 TAP 算法与公式(1)中的 AP 算法可以发现,TAP 算法借鉴了源域中的数据信息,具备了迁移学习能力,λ_1 的大小体现了对目标域潜在类中心点的附加惩罚值.另外,通过对源域聚类代表点近邻内的数据点增加惩罚项 $\Delta_k(c)=\lambda_2\cdot I$, if $c_k=k$ and $(\exists j,c_k\in neb(c'_j))$,缩小了 AP 算法搜索最终聚类代表点的范围,同时增大了源域近邻内数据成为最终类代表点的可能性.因此,TAP 在目标域的数据或信息不足的情况下,可以利用源域中的信息来帮助其学习,而且可以加快算法的收敛速度,从而可以更容易地找到目标域数据的聚类代表点和分配矩阵.图 4 为 TAP 算法的流程.

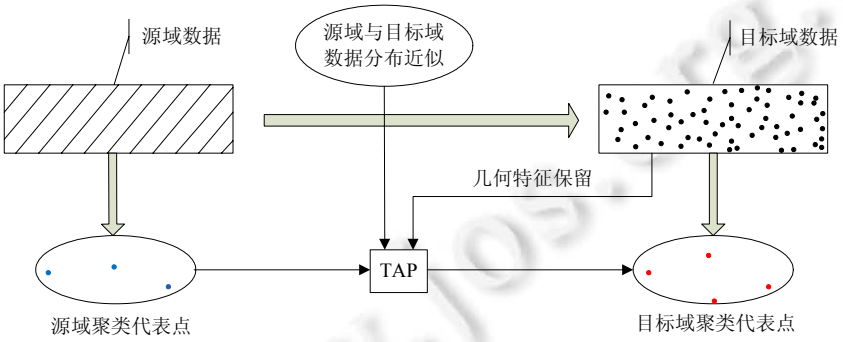


Fig.4 Diagram of TAP method

图 4 TAP 算法流程图

2.3 TAP优化处理

与 AP 算法的优化过程类似,通过最大化目标函数公式(10)来寻求最优分配矩阵是一个 NP 难题[1].本文中,将继续利用因子图的信念传播和最大化的技巧来解决此问题,以保证算法收敛于领域最大值[2].本文算法对应的因子图为图 5,在该因子图中,目标函数 $S(c)$如公式(10)中给出,其中,Δ_i 为新的惩罚函数.如果数据 i 是选自源域

用来保持分布一致性,则 $s'(i,j)=\lambda_1\cdot[s(i,j)-s(i,c_i')]^2$;否则,$s'(i,j)=0$.$s'(i,j)$体现了对目标域数据和源域数据的分布一致性约束,其值越大,则第 j 个点成为代表点的概率越大.c_i 和 $s(i,\cdot)$则类似于 AP,分别表示潜在聚类代表点与数据点间的负欧式距离.

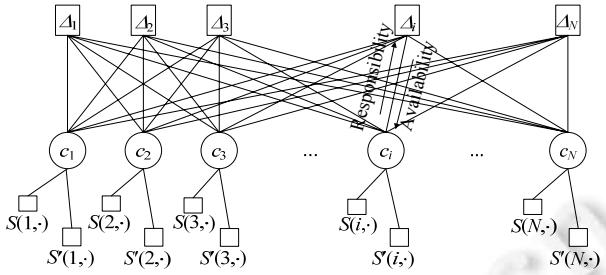


Fig.5 Factor graph of the TAP method

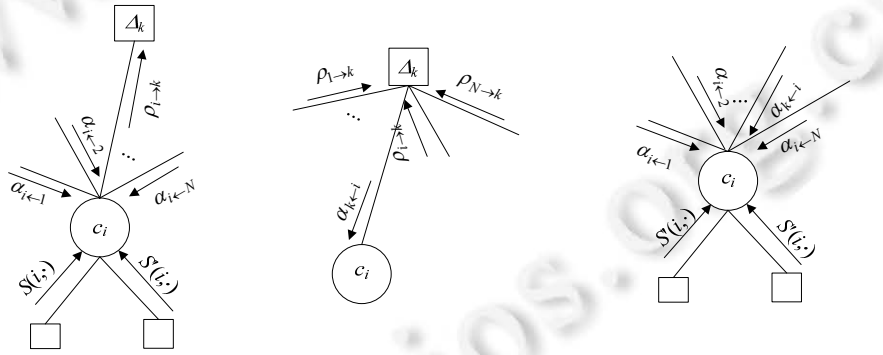
图 5 TAP 方法的因子图

2.3.1 消息传递

与 AP 类似,TAP 聚类代表点的确定也是双向选择过程,由潜在类代表点传递给惩罚项的消息为ρ,由惩罚项传递给潜在代表点的消息表示为α.

如图 6(b)所示,从潜在类代表点 c_i 传递给惩罚项$\Delta_k(c)$的信息由 N 个消息组成,表示为$\rho_{i\leftarrow k}(j)$.

$$\begin{aligned}\rho_{i\to k}(c_i)&=\overbrace{s(i,c_i)+s'(i,c_i')+[\exists j,c_i\in neb(c_j')]\cdot\lambda_2\cdot I}^{1\ message}+\overbrace{\sum_{k':k'\neq k}\alpha_{i\leftarrow k'}(c_i)}^{N-1\ messages}\\&=\overbrace{S_{ici}(q_i)+[\exists j,c_i\in neb(c_j')]\cdot\lambda_2\cdot I}^{1\ message}+\overbrace{\sum_{k':k'\neq k}\alpha_{i\leftarrow k'}(c_i)}^{N-1\ message}\\&=\overbrace{s(i,c_i)+\lambda_1\cdot[s(i,c_i)-s(i,c_i')]^2\cdot[q_i]+[\exists j,c_i\in neb(c_j')]\cdot\lambda_2\cdot I}^{1\ message}+\overbrace{\sum_{k':k'\neq k}\alpha_{i\leftarrow k'}(c_i)}^{N-1\ message}\end{aligned}\tag{11}$$



(a) 点 c_i 的消息传递 (b) 从点 c_i 传递到Δ_k的消息$\rho_{i\leftarrow k}$ (c) 从Δ_k传递到点 c_i 的消息$\alpha_{i\leftarrow k}$

Fig.6 Factor graph of messages of TAP method

图 6 TAP 算法消息传递因子图

另一方面,从惩罚项$\Delta_k(c)$传递给潜在代表点 c_i 的消息α通过除了当前类代表点之外的所有流入消息和自身惩罚总共 N 个值求和并最大化.如图 6(c)所示,该值表示为$\alpha_{i\leftarrow k}(j)$.

$$\alpha_{i\to k}(c_i)=\overbrace{\max_{j_1,\ldots,j_{i-1},j_{i+1},\ldots,j_N}\left[\Delta_k(j_1,\ldots,j_{i-1},c_i,j_{i+1},\ldots,j_N)+\sum_{i'}\rho_{i'\to k}(j_{i'})\right]}^{best\ possible\ configuration\ satisfying\ \Delta_k\ given\ c_i}\tag{12}$$

2.3.2 简化后的传递消息和分配矩阵

与 AP 的计算策略相似,我们亦可对传递消息及分配矩阵进行简化,公式(11)~公式(15)的具体推导详见本文的附录.可以得到:

$$r(i,k)=s(i,k)+s'(i,k)+[k\in neb(c'_k)]\cdot\lambda_2\cdot I-\max_{j:j\neq k}[s(i,j)+s'(i,j)+a(i,j)] \tag{13}$$

$$a(i,k)=\begin{cases}[k\in neb(c'_k)]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval\\ i'\neq k}}(r(i',k)+\lambda_2\cdot I)+\sum\limits_{i'':i''\neq i',k}\max(0,r(i'',k)), & k=i\\ \min\left[0,[k\in neb(c'_k)]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval\\ i'\neq i,k}}(r(i',k)+\lambda_2\cdot I)+\sum\limits_{i'':i''\neq i',k}\max(0,r(i'',k))\right], & k\neq i\end{cases} \tag{14}$$

上式中,*maxval* 表示源域数据到其对应代表点的最大距离,也可以由使用者自己指定.这里,λ_2 的作用类似于 AP 算法中的“preference”.值得注意的是,后者针对整个数据集,其绝对值越大,聚类个数越少;前者针对的是那些落在代表点的 *maxval* 近邻内的数据点.

从公式(13)、公式(14)可以看出,新的 $r(i,k)$除了保留经典 AP 中 $r(\cdot,\cdot)$的功能以外,还进一步考虑了代表点的合适度问题,如,若第 k 个点在源域代表点近邻内且其分布与源域数据相似,则第 k 个点适合作为第 i 个样本的类代表点的可能性会增大;同样,新的 $a(i,k)$也在保留经典 AP 算法中 $a(\cdot,\cdot)$的功能以外,进一步考虑了代表点的选择性问题,如,若第 k 个点在源域代表点的近邻内,那么第 i 个点选择第 k 个点作为类代表点的可能性会增大.

矩阵 $\boldsymbol{r},\boldsymbol{a}$ 初始化为 **0** 矩阵,类似于常见的迭代优化算法.本文算法的终止条件亦采用如下两种策略之一:(1) 固定迭代次数;(2) 算法收敛.为了得到 c_i 的值,将所有流入 c_i 的值求和以及最大值,可以得到聚类结果 $\hat{c}_i$.

$$\hat{c}_i=\arg\max_j[a(i,j)+r(i,j)] \tag{15}$$

2.4 TAP全局收敛性分析

TAP 是 AP 的泛化版本,AP 是一种不参考任何外部有用信息进行聚类的算法,而 TAP 在分布和类代表点方面参考了源域信息,若将 TAP 中的分布参考系数λ_1 和近邻个数设置为 0,则 TAP 算法将退化为 AP 算法.另外,由于 TAP 利用源域信息对目标域中近邻范围内的潜在类代表点施加了更大的权值,其收敛速度比 AP 更快.值得注意的是,TAP 和 AP 一样,当数据迭代过程中出现多个使全局最小的等价类代表点时,算法将不再收敛.这时,消息的传递将会出现震荡,用μ表示公式(13)和公式(14)的左边部分,通过衰变系数η和 $1-\eta$该对迭代过程中的上一次与本次结果进行线性组合,如下式:

$$\mu=\eta\cdot\mu^{old}+(1-\eta)\cdot\mu^{new} \tag{16}$$

该方法在一定程度上可以避免震荡效应.需要说明的是,TAP 使用了和 AP 算法同样的优化策略,但 TAP 缩小了选择潜在类代表点的搜索区域,本质上增大了某些更可能成为类代表点的潜在代表点的概率,从而排除了部分使得全局最小的同等代表点,避免了很多不必要的震荡而加快了收敛速度.这里,η的取值参考 AP 算法.

从时间复杂度来看,AP 算法的时间复杂度为 $O(n^2)$,本文算法的时间复杂度为 $O(2n^2)$.分析如下:在迭代之前,本文算法需要估算出源域类代表点在目标域的近邻,其时间复杂度为 $O(n^2)$,在缩小了潜在类代表点的范围后,聚类过程类似于 AP.

3 实验研究

3.1 实验设置

为了验证本文方法在数据匮乏等复杂情况下的聚类性能,本节将分别通过人工合成数据集以及真实的网络入侵检测数据 KDD99 和 SEA 数据集来对 TAP 算法进行分析与评估.有关人工合成数据以及真实数据的详细描述将分别于第 3.2 节及第 3.3 节中给出.此外,为对本文所提出的 TAP 算法聚类性能做出评判,将于第 3.2 节和第 3.3 节中给出与当前最新的迁移谱聚类算法(transfer spectral clustering,简称 TSC)[21]、AP 算法[1]、基于真实类中心的 *k*-centers 聚类算法[1]、直接采用源域数据聚类获得的类代表点来标注目标域数据的方法(LT)以

及当前较流行的一种数据流聚类算法(data stream clustering with affinity propagation,简称 StrAP)[27]在人工集和真实集上进行性能比较,并对此结果进行适当的分析与解释.各算法的参数设置如下:

1) 在 AP 和 TAP 中,当聚类结果保持 100 次不变时算法终止,即,t_{conv}=100,最大迭代次数设置为 $t_{\max}$=1000,相似度的计算利用负欧式距离.在[1,…,7]内使用网格搜索技术确定最佳近邻个数,λ_1 的取值范围为[0.1,0.2,…1],λ_2 的取值范围为[1,2,…10].

2) *k*-centers 和 TSC 算法的参数设置参见文献[1,21].

对于 StrAP 算法,我们将不同的数据片段当作数据流处理,且在当前时刻对聚类中心进行更新以便于处理下一片段的数据.对于本文算法,为利用公式(5-2)检验源域和目标域的分布是否一致,以实现有效迁移.在实验过程中,需随机抽取 10%的源域样本用于上述检验.此外,该部分抽取的样本将作为源域对目标域的辅助知识参与目标域的聚类过程.

为了公正地对各聚类算法的聚类性能做出合理的评价,本文采用 3 种评价指标进行算法的性能分析.

1) 精度 ACC[28,29],定义如下:

$$ACC=\frac{\sum_{i=1}^{N}\delta(y_i,map(c_i))}{N} \tag{17}$$

其中,N 是数据点的个数,y_i 和 c_i 分别表示真实数据标签和所获得的聚类标签.$\delta(y,c)$表示为:当 $y=c$ 时,函数值为 1,否则为 0.$map(\cdot)$是一个排列函数,它将每一个聚类标签和类标进行匹配,最优匹配结果详见 Hungarian 算法[30].

2) 标准化互信息 NMI[31],定义如下:

$$NMI=\frac{\sum_{i=1}^{C}\sum_{j=1}^{C}N_{i,j}\log\frac{N\cdot N_{i,j}}{N_i\cdot N_j}}{\sqrt{\sum_{i=1}^{C}N_i\log\frac{N_i}{N}\cdot\sum_{j=1}^{C}N_j\log\frac{N_j}{N}}} \tag{18}$$

其中,$N_{i,j}$ 表示聚类结果第 i 类中和真实标签第 j 类中共同数据的个数,N_i 表示在类 i 中数据的个数,N_j 表示在类 j 中数据的个数,N 表示整个数据集中数据的个数.

3) 芮氏指标 RI[32,33],定义如下:

$$RI=\frac{f_{00}+f_{11}}{N(N-1)/2} \tag{19}$$

其中,f_{00} 表示数据点具有不同的类标签并且属于不同类的配对点数目,f_{11} 表示数据点具有相同的类标签并且属于同一类的配对点数目,N 表示整个数据样本的总量大小.

以上 3 种方法,取值范围均为[0,1],且均随着数值的偏高,显示出算法的性能更为优越.

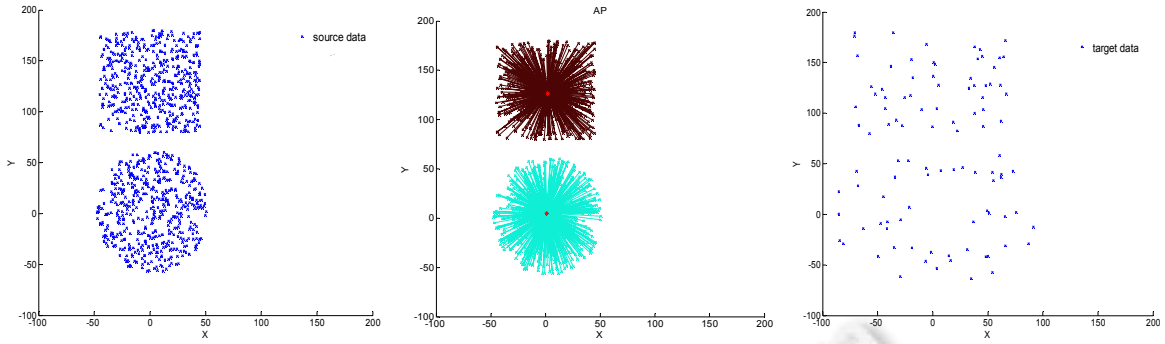
实验环境:实验硬件平台为 Windows 32 位 4 Intel Core i3,内存为 4GB.编程环境为 MATLAB 2012b.

3.2 人工合成数据集实验

为了充分验证本文方法在目标域场景数据量不足或所含信息匮乏时的聚类效果,在人工合成数据集部分,本文首先构造具有充分的数据的源域数据集,如图 7(a)所示,用正方形和圆形来表示源域的事物/对象,而后构造数据不足的目标域数据集;如图 7(c)所示,用长方形和椭圆形来表示目标域的事物/对象.对于源域而言,包含两类样本,每一类样本由 500 个点对构成共 1 000 个样本.目标域数据量占源域的 10%,如图 7 所示.

由于本文所采用的迁移策略,在 TAP 算法中,我们从源域中随机抽出 10%数据,用于检验领域分布一致性.此外,在实验过程中,我们先将数据归一化后再对其进行聚类.首先使用 AP 算法对于源域数据进行聚类,通过调节“preference”中值得到如图 7(b)所示的聚类结果,其中,用红色标出的数据为正方形和圆形的类代表点.

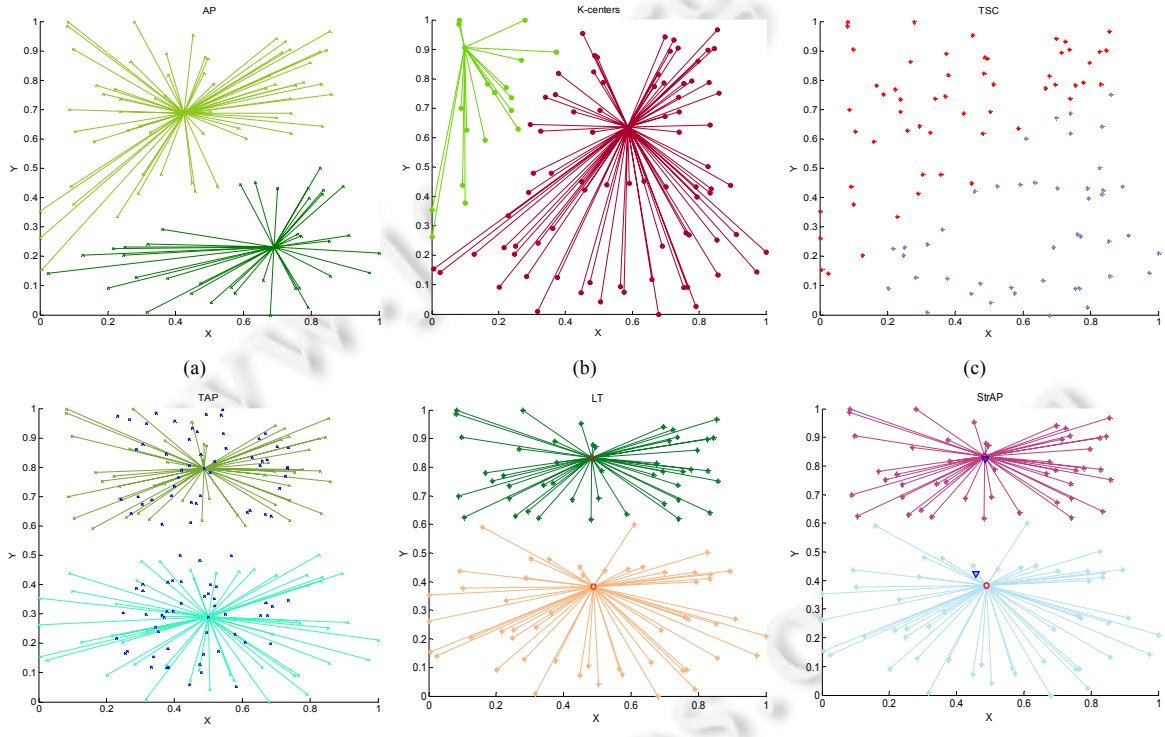
图 8 和表 1 展示了 TAP 算法与其余 3 种对比聚类算法在模拟目标域数据集上的聚类结果.



(a) 源域数据集 (b) AP 对源域数据集聚类效果 (c) 目标域数据集

Fig.7 Source dataset and the corresponding insufficient target datasets

图 7 源域数据集及其对应的非充分目标域数据集



(a) (b) (c)

(d) (e) (f)

Fig.8 Clustering performance on target dataset obtained by TAP and other comparison algorithms

图 8 TAP 及对比算法在目标数据集上的聚类效果

Table 1 Comparison of clustering indices NMI, ACC and RI of several algorithms on the synthetic datasets

表 1 各种算法在模拟数据集上的聚类指标 NMI,ACC 以及 RI 的比较

Dataset	Index	LT	AP	*K*-Centers	TSC	StrAP	TAP
图 7(c)	NMI	0.878 2	0.572 5	0.070 2	0.297 7	0.878 2	**1.000 0**
	ACC	0.970 0	0.880 0	0.519 3	0.806 5	0.970 0	**1.000 0**
	RI	0.960 4	0.786 6	0.610 0	0.688 2	0.960 4	**1.000 0**

通过对相关实验结果的分析,可以得到以下结论:

(1) 图 8 是 4 种算法在目标数据集上的聚类效果,对于 *K*-Centers 和 TSC 算法,将聚类个数 2 作为输入参数.

从图 8(a)~图 8(c)可以看出,在处理目标域数据匮乏的聚类任务时,若继续使用原始 AP 或 *K*-Centers 聚类算法,其聚类性能明显不佳.造成这种现象的主要原因是,由于数据量少,导致当前数据所包含的信息不足,从而增加了有效聚类的难度,使得在继续使用原方法聚类时,聚类中心偏离实际聚类代表点,导致聚类效果变差.而基于谱聚类的迁移谱聚类算法(TSC)算法同样也未得到较好的聚类效果,这是由于该迁移聚类方法在知识迁移时仅考虑了数据的统计特征而未考虑数据的几何结构,当数据量不足时,分布估计失真影响了算法的迁移质量,从而得不到理想的聚类效果.与 TSC 算法不同的是,本文算法既考虑了源域与目标域数据之间的几何结构特征,又利用统计特征来完成迁移学习,保证了迁移的有效性,使得本文方法取得了理想的聚类效果.

(2) 图 8(d)中的蓝色数据点即是随机选自源域数据集中,用于度量源域与目标域的分布相似程度.综合图 8(a)~图 8(c)可以看出,许多传统基于聚类中心点的聚类算法所发掘的聚类代表点都是虚拟点,而 TAP 与 AP 一样,所发掘的聚类代表点都是数据集中的真实数据点,本质上更具代表性.

(3) 由图 8(e)、图 8(f)及表 1 中有关 LT 和 StrAP 的聚类性能可以发现,LT 和 StrAP 算法与传统的非迁移方法相比有着明显的性能提升,但与本文方法相比还有一定的差距.由于 LT 和 StrAP 算法在聚类过程中均使用了源域信息,如 LT 算法直接使用了源域的类代表中心,而 StrAP 算法也在对目标域聚类时借用了源域的信息,即类代表中心,因此上述两种方法均具备一定的迁移效果.但上述两种方法在聚类过程中均未考虑到源域和目标域的数据分布并非完全一致,其存在一定的差异性.因此,如果 LT 算法直接使用源域类代表中心,其效果必定受限于源域和目标域的数据分布特征.此外,对于 StrAP 算法,由于迁移仅考虑了源域和目标域,其相当于在数据流聚类过程中仅考虑了一个时刻,这样通过 StrAP 算法中的阈值近邻原则[27],针对目标域进行聚类的过程,实际上与 LT 方法相同,其结果也是一样的,图 8(f)中蓝色点即为模型更新之后的中心点.

(4) 表 1 更为具体地展示了 4 种算法在目标域上的聚类性能,从 NMI,AC 和 RI 几个有效性指标来看,TAP 算法都具有明显的性能优势,在所采用的迁移策略的保证下,能够准确无误地将数据分类.与非迁移的 *K*-Centers 算法及经典的 AP 算法相比,由于本文算法能够有效利用源域知识来指导目标域的聚类过程,从而能够获取更为优越的性能.与迁移聚类算法 TSC 算法相比,本文算法同时考虑了几何结构特征和统计特征,从而获取了更为合理的聚类效果,提高了算法的适用性.

综上,本文所提出的 TAP 算法更具迁移学习能力,可获得更佳的聚类性能.此外,由于 TAP 算法继承了 AP 算法的优势,与 *K*-Centers 及 TSC 算法相比,无需预先设定聚类中心数目,而可以通过自身的学习机制自动获取,更具有实用价值.

3.3 真实数据集实验分析

为了对 TAP 算法的聚类性能及实际应用价值作进一步的探讨与分析,本小节将在真实数据集上对 TAP 算法进行探讨.我们选用了两个经典的数据集,即网络入侵检测数据(KDD99)[34]和 SEA[35]数据集.这两个数据集不同的时间片段具有不同的分布,且类内、类间变化较大,对于检测聚类算法的鲁棒性具有实际意义.其中,

(1) KDD99 数据集是一个网络连接定义为在某个时间内从开始到结束的 TCP 数据包序列,并且在这段时间内,数据在预定义的协议下(如 TCP,UDP)从源 IP 地址到目的 IP 地址的传递.由于攻击事件在时间上有很强的关联性,因此统计出当前连接记录与之前一段时间内的连接记录之间存在的某些联系,可以更好地反映连接之间的关系,具有很强的迁移特性.该数据集有 41 个特征,在除去不连续的属性后仅剩下 32 个属性.我们抽取前两周左右,共 10 万数据,该数据块总共有 22 类,其中,Normal 类占 19.621%,Smurf 类占 57.015%,Neptune 类占 21.582%,其余类别总共为 1.782%.

(2) SEA 数据集是一种具有突发性概念漂移特性的数据集,由 Street 等人在 2001 年提出.该数据具有良好的可迁移性,其具有 60 000 个样本、3 个属性,其中两个属性为相关属性,属性值介于 0~10 之间,共包括 4 个概念,每个概念包括 15 000 个样本.除了 10%的噪声点以外,概念函数利用阈值将数据分两大类,大于某个阈值则为第 1 类.表 2 给出了算法测试的数据片段信息.

(3) 在该实验部分,目标域数据量均为源域数据量的 25%,以构成数据匮乏的迁移场景.对于 KDD 数据集而言,除了 3 大类点之外的数据比例极低,可以将其看作噪声点.因此,对于实验中的 *K*-Centers 和 TSC

而言,将类别数设为 3,且将两个数据集均数据归一化.在此基础上,对比各算法的聚类性能.所有实验设置的数据集均运行 10 次,取平均值并给出方差.

Table 2 Source and target datasets generated from KDD99 and SEA

表 2 取自 KDD99 和 SEA 数据集的源域和目标域数据片段

Datasets	Source datasets	Target datasets	Source datasets	Target datasets
KDD'99	1~3000	3001~3750 4001~4750 5001~5750 6001~6750 7001~7750	20001~23000	30001~30750 40001~40750 50001~50750 60001~60750 70001~70750
SEA	1~2000	2001~2500 3001~3500 4001~4500 5001~5500 6001~6500	10001~12000	20001~20500 30001~30500 40001~40500 50001~50500 –

表 3 和表 4 是对 KDD99 数据集的实验结果,其中,表 3 是源域数据采用 1~3000、目标域数据为间隔 1 000 时的实验结果;表 4 是源域数据采用 20001~23000 时,目标域数据为间隔 10 000 时的实验结果.

Table 3 Clustering performances of different algorithms on KDD99 dataset of source dataset 1~3000 and different target datasets

表 3 各种算法对不同目标域数据片段且源域数据为 1~3000 的 KDD99 数据集上的聚类性能比较

Datasets	Index	LT	AP	*K*-Centers	TSC	StrAP	TAP
3001~3750	NMI	0.8475±0	0.5857±0	0.6771±0.1397	0.6561±0.0033	0.8475±0	**0.8561±0**
	AC	0.9427±0	0.7493±0	0.7955±0.1122	0.5375±0.0085	0.9427±0	**0.9440±0**
	RI	0.9571±0	0.7423±0	0.8285±0.0950	0.7675±0.0048	0.9571±0	**0.9593±0**
4001~4750	NMI	0.7954±0	0.5920±0	0.7159±0.1047	0.6792±0.1047	**0.8493±0**	0.8377±0
	AC	0.9240±0	0.7507±0	0.8356±0.0899	0.5913±0.0227	**0.9520±0**	0.9333±0
	RI	0.9395±0	0.7489±0	0.8577±0.0870	0.7914±0.0008	**0.9630±0**	0.9548±0
5001~5750	NMI	0.7444±0	0.5992±0	0.7570±0.0921	0.6531±0.0036	0.8139±0	**0.8483±0**
	AC	0.9093±0	0.7653±0	0.8888±0.0760	0.5700±0.0017	0.9360±0	**0.9440±0**
	RI	0.9099±0	0.7591±0	0.8908±0.0988	0.7740±0.0001	0.9510±0	**0.9639±0**
6001~6750	NMI	0.5802±0	0.6097±0	0.7260±0.1757	0.6615±0.0214	0.8280±0	**0.8468±0**
	AC	0.6880±0	0.7573±0	0.8544±0.1203	0.5807±0.0527	0.9400±0	**0.9440±0**
	RI	0.7670±0	0.7496±0	0.8613±0.1367	0.7696±0.0117	0.9559±0	**0.9625±0**
7001~7750	NMI	0.4676±0	0.5873±0	0.7288±0.1121	0.6588±0.0083	0.8266±0	**0.8590±0**
	AC	0.5533±0	0.7680±0	0.8525±0.0833	0.5755±0.0194	0.9427±0	**0.9493±0**
	RI	0.6761±0	0.7505±0	0.8358±0.1139	0.7835±0.0171	0.9568±0	**0.9680±0**

Table 4 Clustering performances of different algorithms on KDD99 dataset of source dataset 20001~23000 and different target datasets

表 4 各种算法对不同目标域数据片段且源域数据为 20001~23000 的 KDD99 数据集上的聚类性能比较

Datasets	Index	LT	AP	*K*-Centers	TSC	StrAP	TAP
30001~30750	NMI	0.4391±0	0.5930±0	0.7126±0.0893	0.6780±0.0107	0.4391±0	**0.8286±0**
	AC	0.7533±0	0.7693±0	0.8183±0.0598	0.5851±0.0285	0.7533±0	**0.9293±0**
	RI	0.6679±0	0.7718±0	0.8554±0.0781	0.7816±0.0070	0.6679±0	**0.9568±0**
40001~40750	NMI	0.4544±0	0.5833±0	0.5538±0.0975	0.6399±0.0012	0.6138±0	**0.8336±0**
	AC	0.7333±0	0.7440±0	0.7161±0.0750	0.5245±0.0003	0.7720±0	**0.9347±0**
	RI	0.6608±0	0.7423±0	0.7479±0.0739	0.7627±0.0004	0.7390±0	**0.9577±0**
50001~50750	NMI	0.6198±0	0.5987±0	0.7425±0.0898	0.6662±0.0058	0.6933±0	**0.8496±0**
	AC	0.8013±0	0.7613±0	0.8004±0.0807	0.5433±0.0004	0.8893±0	**0.9413±0**
	RI	0.7615±0	0.7470±0	0.8714±0.0810	0.7652±0.0019	0.8812±0	**0.9637±0**
60001~60750	NMI	0.4901±0	0.5937±0	0.7060±0.0836	0.6325±0.0079	0.7228±0	**0.8445±0**
	AC	0.7520±0	0.7733±0	0.7601±0.1087	0.5077±0.0018	0.9000±0	**0.9427±0**
	RI	0.6850±0	0.7645±0	0.8443±0.0763	0.7562±0.0013	0.8949±0	**0.9601±0**
70001~70750	NMI	0.4806±0	0.5900±0	0.7014±0.0883	0.6522±0.0033	0.6440±0	**0.9260±0**
	AC	0.7680±0	0.7827±0	0.8597±0.0553	0.5471±0.0032	0.8000±0	**0.9773±0**
	RI	0.6884±0	0.7565±0	0.8394±0.0721	0.7637±0.0011	0.7521±0	**0.9866±0**

表 5 和表 6 是对 SEA 数据集的测试结果,其中,表 5 是源域数据采用 1~2000、目标域数据间隔 1 000 时的实验结果;表 6 是源域数据采用 10001~12000 时,目标域数据为间隔 10 000 时的实验结果.

Table 5 Clustering performances of different algorithms on SEA dataset of source dataset 1~2000 and different target datasets

表 5 各种算法对不同目标域数据片段且源域数据为 1~2000 的 SEA 数据集上的聚类性能比较

Datasets	Index	LT	AP	*K*-Centers	TSC	StrAP	TAP
2001~2500	NMI	0.1996±0	0.0205±0	0.1983±0	0.0091±0	0.1996±0	**0.2172±0**
	AC	0.7360±0	0.5760±0	0.7500±0	0.5582±0	0.7360±0	**0.7540±0**
	RI	0.6106±0	0.5106±0	0.6242±0	0.5079±0	0.6106±0	**0.6283±0**
3001~3500	NMI	0.1015±0	0.0009±0	0.0273±0.0193	0.0322±0	0.1935±0	**0.2180±0**
	AC	0.5500±0	0.5380±0	0.6124±0.0180	0.5933±0	0.7280±0	**0.7300±0**
	RI	0.5040±0	0.5019±0	0.5249±0.0090	0.5187±0	0.6032±0	**0.6050±0**
4001~4500	NMI	0.0971±0	0.0056±0	0.0988±0.0507	0.0200±0.0002	**0.1664±0**	0.1595±0
	AC	0.5840±0	0.5420±0	0.6772±0.0765	0.5849±0.0001	**0.6820±0**	0.6780±0
	RI	0.5131±0	0.5025±0	**0.5725±0.0427**	0.5147±0.0001	0.5654±0	0.5625±0
5001~5500	NMI	0.1042±0	0.1582±0	0.0507±0.0334	0.0428±0	0.1552±0	**0.1725±0**
	AC	0.6040±0	0.7000±0	0.5810±0.0896	0.6143±0	0.6840±0	**0.7200±0**
	RI	0.5207±0	0.5792±0	0.5266±0.0419	0.5262±0	0.5668±0	**0.5960±0**
6001~6500	NMI	0.0987±0	0.0004±0	0.1008±0	0.0538±0	0.1916±0	**0.2617±0**
	AC	0.5740±0	0.5280±0	0.5478±0	0.6248±0	0.7180±0	**0.7880±0**
	RI	0.5100±0	0.5006±0	**0.8394±0**	0.5312±0	0.5942±0	0.6652±0

Table 6 Clustering performances of different algorithms on SEA dataset of source dataset 10001~12000 and different target datasets

表 6 各种算法对不同目标域数据片段且源域数据为 10001~12000 的 SEA 数据集上的聚类性能比较

Datasets	Index	LT	AP	*K*-Centers	TSC	StrAP	TAP
20001~20500	NMI	0.1107±0	**0.1982±0**	0.1856±0.0897	0.0272±0	0.1107±0	0.1859±0
	AC	0.6160±0	**0.7540±0**	0.7364±0.0734	0.5973±0	0.6160±0	0.7440±0
	RI	0.5260±0	**0.6283±0**	0.6207±0.0606	0.5192±0	0.5260±0	0.6244±0
30001~30500	NMI	0.0897±0	0.1270±0	0.0749±0.0462	0.0277±0.0002	0.0904±0	**0.1315±0**
	AC	0.5900±0	0.6960±0	0.6348±0.0449	0.6000±0.0014	0.5640±0	**0.6980±0**
	RI	0.5152±0	0.5760±0	0.5390±0.0233	0.5260±0.0006	0.5072±0	**0.5994±0**
40001~40500	NMI	0.0486±0	0.0504±0	3.5713e−6±0	0.0364±0.0003	0.0506±0	**0.0643±0**
	AC	0.5640±0	**0.6460±0**	0.5640±0	0.6105±0.0007	0.5540±0	0.6180±0
	RI	0.5072±0	**0.5417±0**	0.5072±0	0.5244±0.0003	0.5048±0	0.5269±0
50001~50500	NMI	0.0886±0	0.1373±0	0.0126±0.0054	0.0399±0	0.1221±0	**0.1732±0**
	AC	0.5500±0	0.7060±0	0.5552±0.0896	0.6115±0	0.5680±0	**0.7220±0**
	RI	0.5040±0	0.5840±0	0.5051±0.0006	0.5250±0	0.5083±0	**0.5978±0**

观察表 3~表 6 的实验结果,我们可以得到如下结论:

(1) 表 3~表 6 的结果显示:在大多数情况下,TAP 算法在 NMI,ACC 及 RI 这 3 个聚类有效性指标上均优于其他算法.这进一步说明了 TAP 算法在源域数据知识的辅助下,提升了对目标域数据的聚类效果.值得注意的是,迁移聚类算法 TSC 本质上是采用了一种任务数为 2 的多任务学习机制来完成迁移学习,其在谱聚类的基础上,通过一种协调机制对源域和目标域数据进行协同,在数据的分布特征类似时,具有同时提高双方学习性能的作用.但是对于一类数据较少导致几何特征分布发生变化的数据而言,该算法显得并不是非常有效.

(2) 从 4 张实验结果表中可以看出:以 AP 为基础的一类聚类算法对数据的初始化不敏感,这使得 AP 以及 TAP 算法的聚类结果方差为 0.可以看出,TAP 算法继承了 AP 算法稳定性的优良特征.相对于其他聚类算法而言,更具实用价值.

(3) 表 6 第 1 个目标域数据片段的实验结果表明,AP 算法的聚类结果要优于 TAP 算法.这说明对源域聚类代表点近邻内的目标域潜在聚类代表点增加权重是不合理的,导致这种情况的原因是,由于此段目标域数据域源域相差很大,而且真实的聚类代表点已经不在源域聚类代表点的近邻内,从而会出现负

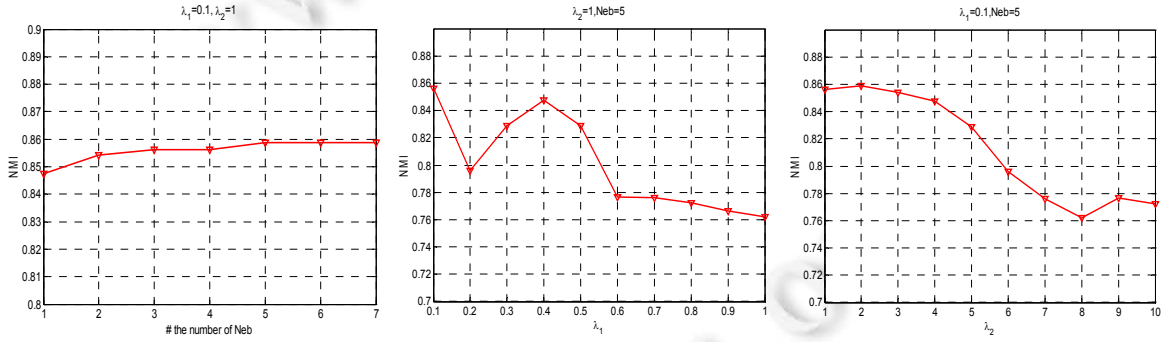
迁移的现象.需要说明的是,此类情况的发生表明源域与目标域数据发生较大的偏差,这也是当前大部分迁移学习算法所面临的问题.在目标域为 40001~40500 时,各种算法得到的实验结果均不理想,但 TAP 算法仍比其他算法有更好的聚类结果.

(4) 从表 3~表 6 可以看出:基于源域数据集类代表点打标签的方式在数据结构发生变化时已不再适用,聚类性能也随着这种变化的加剧而下降得越来越明显.而一类基于数据流的聚类算法通过不断更新聚类模型[27]来保持聚类性能,这往往通过一些知识保留技术(例如,通过衰变函数机制)来达到更新聚类模型的目的.值得注意的是,在当前的迁移场景下,数据片段是不具连续性的.如表 2 所示,此时使用数据流算法,其聚类模型由于上下时刻数据存在断层而导致聚类中心得不到准确的更新,从而达不到预期的聚类效果.

综上,通过在真实数据集上的实验与分析,我们可以得到一个较明确的结论,即 TAP 算法在处理数据匮乏情况下的聚类任务时一般均优于非迁移聚类算法,而同时考虑分布特性和几何特征的 TAP 算法又要优于以往的迁移聚类算法.至此,本文算法的优越性能得到了充分的验证与肯定.

3.4 参数敏感性分析

为了进一步考量本文所涉及的预设参数在具体的聚类过程中对算法最终聚类性能所产生的影响,本节将以 KDD99 数据集为例对算法参数的敏感性进行分析.具体来说,以 KDD99 数据集中的数据片段 1~3000 作为源域,数据片段 3001~3750 作为目标域.由于本文方法共包含 3 个人工设定参数,即参数λ_1,λ_2 以及近邻个数,因此,我们采用将两个参数固定到各自对应的最优值,改变另一参数来观察算法的性能变化.图 9 展示了 3 个参数对本文算法聚类性能的影响.由图中结果可知,近邻个数的变化对本文算法的影响较小,其性能变化趋势最为平缓稳定,而参数λ_1 及λ_2 控制着本文算法的迁移程度.图 9 的结果说明了其数值的变化会对本文算法的聚类性产生一定的影响,但该影响也是在可接受范围内的.综上,本文算法性能在各参数的影响下,其所得结果较为稳定,参数敏感性不大.



(a) 近邻个数对 TAP 算法的影响 (b) 参数λ_1 对 TAP 算法的影响 (c) 参数λ_2 对 TAP 算法的影响

Fig.9 Influence of parameter λ_1, λ_2 and the number of neighbors with different values for the clustering performances

图 9 参数λ_1,λ_2 以及近邻个数取不同值时对聚类效果的影响

4 结 论

本文针对目标域数据样本匮乏导致传统数据分析任务失效的问题,在经典 AP 算法的基础上引入迁移学习机制,利用源域知识辅助目标域学习,提出一种迁移近邻传播聚类算法 TAP.该算法是一种基于数据分布和聚类代表点几何结构的知识迁移聚类算法,既利用了源域数据的几何结构特征,又利用了其统计特征,进而得到更具指导意义的全局性聚类划分结果.TAP 是 AP 的泛化版本,其以一种类似 AP 算法的因子图的信息传播方法自行识别聚类个数,并得到相应的分配矩阵.在人工数据和真实数据上的实验结果,反映出 TAP 算法对于领域间知识

迁移学习的有效性和高效性.

尽管 TAP 算法在领域间知识迁移上表现出很高的实用价值,但仍然存在一些值得将来讨论的问题.例如,TAP 仍然采用经典的 AP 框架,对于欧氏距离的使用致使本文算法在面对高维数据聚类问题时算法性能将面临一定的考验.另外,在对不同领域间的知识迁移时,这种领域差异性究竟在什么程度级别才适合迁移?这种程度是如何体现在参数λ_1,λ_2上的?这将成为我们今后研究的重点之一.

References:

[1] Frey BJ, Dueck D. Clustering by passing messages between data points. Science, 2007,315(5814):972−976. [doi: 10.1126/science.1136800]

[2] Kschischang FR, Frey BJ, Loeliger HA. Factor graphs and the sum-product algorithm. IEEE Trans. on Information Theory, 2001, 47(2):498−519. [doi: 10.1109/18.910572]

[3] McQueen JB. Some methods for classification and analysis of multivariate observations. In: Proc. of the 5th Berkeley Symp. on Mathematical Statistics and Probability. Berkeley: University of California Press, 1967. 281−297.

[4] Kohonen T. Self-Organizing Feature Maps. Berlin, Heidelberg: Springer-Verlag, 1989. [doi: 10.1007/978-3-642-88163-3_5]

[5] Dueck D, Frey BJ, Jojic N, Jojic V, Giaever G, Emili A, Musso G, Hegele R. Constructing treatment portfolios using affinity propagation. In: Proc. of the 12th Annual Int'l Conf. on Research in Computational Molecular Biology (RECOMB 2008). 2008. 360−371. [doi: 10.1007/978-3-540-78839-3_31]

[6] Jain AK. Data clustering: 50 years beyond *K*-means. Pattern Recognition Letters, 2010,31(8):651−666. [doi: 10.1016/j.patrec.2009.09.011]

[7] Sumedha ML, Weigt M. Unsupervised and semi-supervised clustering by message passing: Soft-Constraint affinity propagation. The European Physical Journal B, 2008,66(1):125−135. [doi: 10.1140/epjb/e2008-00381-8]

[8] Xiao J, Wang J, Tan P, Quan L. Joint affinity propagation for multiple view segmentation. In: Proc. of the 11th IEEE Int'l Conf. on Computer Vision. IEEE Press, 2007. 1−7. [doi: 10.1109/ICCV.2007.4408928]

[9] Xiao Y, Yu J. Semi-Supervised clustering based on affinity propagation algorithm. Ruan Jian Xue Bao/Journal of Software, 2008, 19(11):2803−2813 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/2803.htm [doi: 10.3724/SP.J.1001.2008.02803]

[10] Pan SJ, Yang Q. A survey on transfer learning. IEEE Trans. on Knowledge and Data Engineering, 2010,22(12):1345−1359. [doi: 10.1109/TKDE.2009.191]

[11] Shao L, Zhu F, Li X. Transfer learning for visual categorization: A survey. IEEE Trans. on Neural Networks and Learning Syatems, 2015,26(5):1019−1034. [doi: 10.1109/TNNLS.2014.2330900]

[12] Zhuang FZ, He Q, Shi ZZ. Survey on transfer learning research. Ruan Jian Xue Bao/Journal of Software, 2015,26(1):26−39 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4631.htm [doi: 10.13328/j.cnki.jos.004631]

[13] Deng ZH, Jiang YZ, Choi KS, Chung FL, Wang ST. Knowledge-Leverage-Based TSK fuzzy system modeling. IEEE Trans. on Neural Networks and Learning Systems, 2013,24(8):1200−1212. [doi: 10.1109/TNNLS.2013.2253617]

[14] Raina R, Battle A, Lee H, Packer B, Ng AY. Self-Taught learning: Transfer learning from unlabeled data. In: Proc. of the 24th Int'l conf. on Machine Learning. ACM Press, 2007. 759−766. [doi: 10.1145/1273496.1273592]

[15] Xue GR, Dai W, Yang Q, Yong Y. Topic-Bridged PLSA for cross-domain text classification. In: Proc. of the 31st Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR 2008). ACM Press, 2008. 627−634. [doi: 10.1145/1390334.1390441]

[16] Glorot X, Bordes A, Bengio Y. Domain adaptation for large-scale sentiment classification: A deep learning approach. In: Proc. of the 28th Int'l Conf. on Machine Learning (ICML). ICML Press, 2011. 513−520.

[17] Dai W, Yang Q, Xue G, Yu Y. Boosting for transfer learning. In: Proc. of the Int'l Conf. on Machine Learning (ICML). ICML Press, 2007. 193−200. [doi: 10.1145/1273496.1273521]

[18] Tommasi T, Orabona F, Caputo B. Learning categories from few examples with multi model knowledge transfer. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2014,36(5):928−942. [doi: 10.1109/TPAMI.2013.197]

[19] Deng ZH, Jiang YZ, Cao LB, Wang ST. Knowledge-Leverage-Based TSK fuzzy system with improved knowledge transfer. In: Proc. of the 2014 IEEE Int'l Conf. on Fuzzy System. IEEE Press, 2014. 178−185. [doi: 10.1109/FUZZ-IEEE.2014.6891544]

[20] Jiang YZ, Deng ZH, Wang J, Qian PJ, Wang ST. Collaborative partition multi-view fuzzy clustering algorithm using entropy weighting. Ruan Jian Xue Bao/Journal of Software, 2014, 25(10):2293−2311 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4510.htm [doi: 10.13328/j.cnki.jos.004510]

[21] Jiang W, Chung F. Transfer spectral clustering. In: Proc. of the Machine Learning and Knowledge Discovery in Databases. Berlin, Heidelberg: Springer-Verlag, 2012. 789−803. [doi: 10.1007/978-3-642-33486-3_50]

[22] Ling X, Dai W, Xue GR, Yang Q, Yu Y. Spectral domain-transfer learning. In: Proc. of the 14th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. ACM Press, 2008. 488−496. [doi: 10.1145/1401890.1401951]

[23] Wang C, Mahadevan S. Manifold alignment without correspondence. In: Proc. of the 21st Int'l Joint Conf. on Artificial Intelligence. 2009. 1273−1278.

[24] Pan SJ, Tsang IW, Kwok JT, Yang Q. Domain adaptation via transfer component analysis. IEEE Trans. on Neural Networks, 2011, 2(22):199−210. [doi: 10.1109/TNN.2010.2091281]

[25] Wang C, Mahadevan S. Heterogeneous domain adaptation using manifold alignment. In: Proc. of the 25th AAAI Conf. on Artificial Intelligence. IJCAI Press, 2011. 1541−1546. [doi: 10.5591/978-1-57735-516-8/IJCAI11-259]

[26] Shi X, Liu Q, Fan W, Yu PS. Transfer across completely different feature spaces via spectral embedding. IEEE Trans. on Knowledge and Data Engineering, 2013,25(4):906−918. [doi: 10.1109/TKDE.2011.252]

[27] Zhang XL, Furtlehner C, Germain-Renaud C, Sebag M. Data stream clustering with affinity propagation. IEEE Trans. on Knowledge and Data Engineering, 2014,26(7):1644−1656. [doi: 10.1109/TKDE.2013.146]

[28] Chen WY, Song Y, Bai H, Chang EY. Parallel spectral clustering in distributed systems. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2011,33(3):568−586. [doi: 10.1109/TPAMI.2010.88]

[29] Wu M, Schölkopf B. A local learning approach for clustering. In: Proc. of the Conf. on Neural Information Processing Systems. MIT Press, 2007. 1529−1536.

[30] Papadimitriou CH, Steiglitz K. Combinatorial Optimization: Algorithms and Complexity. Prentice Hall, Inc., 1998.

[31] Strehl A, Ghosh J. Cluster ensembles—Knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research, 2003,3(3):583−617. [doi: 10.1162/153244303321897735]

[32] Deng ZH, Choi KS, Chung FL, Wang ST. Enhanced soft subspace clustering integrating within cluster and between cluster information. Pattern Recognition, 2010,43(3):767−781. [doi: 10.1016/j.patcog.2009.09.010]

[33] Liu J, Mohammed J, Carter J, Ranka S, Kahveci T, Baudis M. Distance based clustering of CGH data. Bioinformatics, 2006,22(16): 1971 −1978. [doi: 10.1093/bioinformatics/btl185]

[34] Aggarwal CC, Han J, Wang J, Yu P. A framework for clustering evolving data streams. In: Proc. of the 29th VLDB Conf. VLDB Endowment, 2003. 81−92.

[35] Tsymbal A, Pechenizkiy M, Cunningham P, Puuronen S. Dynamic integration of classifiers for handling concept drift. Information Fusion, 2008,9(1):56−68. [doi: 10.1016/j.inffus.2006.11.002]

附中文参考文献:

[9] 肖宇,于剑.基于近邻传播算法的半监督聚类.软件学报,2008,19(11):2803−2813. http://www.jos.org.cn/1000-9825/19/2803.htm [doi: 10.3724/SP.J.1001.2008.02803]

[12] 庄福振,何清,史忠植.迁移学习研究进展.软件学报,2015,26(1):26−39. http://www.jos.org.cn/1000-9825/4631.htm [doi: 10.13328/j.cnki.jos.004631]

[20] 蒋亦樟,邓赵红,王骏,钱鹏江,王士同.熵加权多视角协同划分模糊聚类算法.软件学报,2014,25(10):2293−2311. http://www.jos.org.cn/1000-9825/4510.htm [doi: 10.13328/j.cnki.jos.004510]

附录

根据第 2.3 节中传递的两类消息,其中,从潜在类代表点传出的ρ-消息表示为

$$\rho_{i\to k}(c_i)=s(i,c_i)+s'(i,c_i')+\sum_{k':k'\neq k}\alpha_{i\leftarrow k'}(c_i)+[c_i\in neb(i,c_k')]\cdot\lambda_2\cdot I \tag{20}$$

从惩罚函数传递给潜在代表点的α-消息如下:

$$\alpha_{i\to k}(c_i)=\overbrace{\max_{j_1,\ldots,j_{i-1},j_{i+1},\ldots,j_N}\left[\Delta_k(j_1,\ldots,j_{i-1},c_i,j_{i+1},\ldots,j_N)+\sum_{i'}\rho_{i'\to k}(j_{i'})\right]}^{\text{best possible configuration satisfying }\Delta_k\text{ given }c_i}$$

$$=\begin{cases}\overbrace{[k\in neb(c_k')]\cdot\sum_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq k}}(\rho_{i'\to k}(k)+\lambda_2\cdot I)+\sum_{i'':i''\neq i',k}\max_{j'}\rho_{i''\to k}(j')}^{\text{best configuration with or without cluster }k}, & \text{for } c_i=k=i,\\ \overbrace{\sum_{i':i'\neq k}\max_{j':j'\neq k}\rho_{i'\to k}(j')}^{\text{best configuration with no cluster }k}, & \text{for } c_i\neq k=i,\\ \overbrace{\rho_{k\to k}(k)+[k\in neb(c_k')]\cdot\lambda_2\cdot I}^{k\text{ is an exemplar}}+ & \\ \overbrace{[k\in neb(c_k')]\cdot\sum_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\rho_{i'\to k}(k)+\lambda_2\cdot I)+\sum_{i'':i''\neq i',i,k}\max_{j'}\rho_{i''\to k}(j')}^{\text{best configuration of others}}, & \text{for } c_i=k\neq i\\ \max\left[\begin{array}{l}\overbrace{\max_{j':j'\neq k}\rho_{k\to k}(j')+\sum_{i':i'\neq i,k}\max_{j':j'\neq k}\rho_{i'\to}(j')}^{\text{best configuration with no cluster }k},\\ \overbrace{\rho_{k\to k}(k)[k\in neb(c_k')]\cdot\lambda_2\cdot I+[k\in neb(c_k')]\cdot\sum_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\rho_{i'\to k}(k)+\lambda_2\cdot I)+\sum_{i'':i''\neq i',i,k}\max_{j'}\rho_{i''\to k}(j')}^{\text{best configuration with a cluster }k}\end{array}\right], & \text{for } c_i\neq k\neq i\end{cases} \tag{21}$$

我们可将上述传递的两类消息变量看作一个关于 c_i 的变量和一个常量的和,即

$$\rho_{i\to k}(c_i)=\tilde{\rho}_{i\to k}(c_i)+\bar{\rho}_{i\to k} \tag{22}$$

$$\alpha_{i\leftarrow k}(c_i)=\tilde{\alpha}_{i\leftarrow k}(c_i)+\bar{\alpha}_{i\leftarrow k} \tag{23}$$

于是,上述公式(20)、公式(21)等价于:

$$\rho_{i\to k}(c_i)=s(i,c_i)+s'(i,c_i')+\sum_{k':k'\neq k}\tilde{\alpha}_{i\leftarrow k'}(c_i)+\sum_{k':k'\neq k}\bar{\alpha}_{i\leftarrow k'}(c_i)+[c_i\in neb(c_k')]\cdot\lambda_2\cdot I \tag{24}$$

$$\alpha_{i\to k}(c_i)=\begin{cases}[k\in neb(c_k')]\cdot\sum_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\sum_{i'':i''\neq i',k}\max_{j'}\rho_{i''\to k}(j'), & \text{for } c_i=k=i\\ \sum_{i':i'\neq k}\max_{j':j'\neq k}\tilde{\rho}_{i'\to k}(j')+\sum_{i':i'\neq k}\bar{\rho}_{i'\to k}, & \text{for } c_i\neq k=i\\ \tilde{\rho}_{k\to k}(k)+\bar{\rho}_{k\to k}+[k\in neb(c_k')]\cdot\lambda_2\cdot I+ & \\ [k\in neb(c_k')]\cdot\sum_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\sum_{i'':i''\neq i',i,k}\max_{j'}\rho_{i''\to k}(j'), & \text{for } c_i=k\neq i\\ \max\left[\begin{array}{l}\max_{j':j'\neq k}\tilde{\rho}_{k\to k}(j')+\sum_{i':i'\neq i,k}\max_{j':j'\neq k}\tilde{\rho}_{i'\to k}(j')+\sum_{i':i'\neq i}\bar{\rho}_{i'\to k}\tilde{\rho}_{k\to k}(k)+\bar{\rho}_{k\to k}+[k\in neb(c_k')]\cdot\lambda_2\cdot I+\\ [k\in neb(c_k')]\cdot\sum_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\sum_{i'':i\neq i',i,k}\max_{j'}\rho_{i''\to k}(j')\end{array}\right], & \\ & \text{for } c_i\neq k\neq i\end{cases} \tag{25}$$

当 $c_i\neq k$ 时, $\rho_{i\to k}(c_i)=\tilde{\rho}_{i\to k}(c_i)+\bar{\rho}_{i\to k}$, 我们假设 $\bar{\rho}_{i\to k}=\max_{j:j\neq k}\rho_{i\to k}(j)$, 则有如下结论:

$$\max_{j':j'\neq k}\tilde{\rho}_{i\to k}(j')=0,\max_{j'}\tilde{\rho}_{i\to k}(j')=\max(0,\tilde{\rho}_{i\to k}(k)) \tag{26}$$

类似地,当 $c_i\neq k$ 时, $\alpha_{i\leftarrow k}(c_i)=\tilde{\alpha}_{i\leftarrow k}(c_i)+\bar{\alpha}_{i\leftarrow k}$, 假设 $\bar{\alpha}_{i\leftarrow k}=\alpha_{i\leftarrow k}$, 同样有:

$$\tilde{\alpha}_{i\leftarrow k}(c_i)=0,\sum_{k':k'\neq k}\tilde{\alpha}_{i\leftarrow k'}(c_i)=\tilde{\alpha}_{i\leftarrow c_i}(c_i) \tag{27}$$

对于 $c_i=k$,公式(27)的和为 0.有了公式(26)和公式(27),我们可进一步对公式(24)和公式(25)进行简化:

$$\rho_{i\to k}(c_i)=\begin{cases}s(i,c_i)+s'(i,c_i')+\sum\limits_{k':k'\neq k}\bar{\alpha}_{i\leftarrow k'}(c_i)++[c_i\in neb(c_k')]\cdot\lambda_2\cdot I, & \text{for } c_i=k\\ s(i,c_i)+s'(i,c_i')+\sum\limits_{k':k'\neq k}\tilde{\alpha}_{i\leftarrow k'}(c_i)+\sum\limits_{k':k'\neq k}\bar{\alpha}_{i\leftarrow k'}(c_i), & \text{for } c_i\neq k\end{cases} \tag{28}$$

$$\alpha_{i\to k}(c_i)=\begin{cases}[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\sum\limits_{i'':i''\neq i',k}\max(0,\tilde{\rho}_{i''\to k}(j'))+\\ \sum\limits_{i'':i''\neq i',k}\bar{\rho}_{i''\to k}, & \text{for } c_i=k=i\\ \sum\limits_{i':i'\neq k}\bar{\rho}_{i'\to k}, & \text{for } c_i\neq k=i\\ \tilde{\rho}_{k\to k}(k)+\bar{\rho}_{k\to k}+[k\in neb(c_k')]\cdot\lambda_2\cdot I+[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\\ \sum\limits_{i'':i\neq i',i,k}\max(0,\rho_{i''\to k}(j'))+\sum\limits_{i'':i\neq i',i,k}\bar{\rho}_{i''\to k}, & \text{for } c_i=k\neq i\\ \max\left[\begin{array}{l}\sum\limits_{i':i'\neq i}\bar{\rho}_{i'\to k}\\ \tilde{\rho}_{k\to k}(k)+\bar{\rho}_{k\to k}+[k\in neb(c_k')]\cdot\lambda_2\cdot I+\\ [k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\sum\limits_{i'':i\neq i',i,k}\max(0,\rho_{i''\to k}(j'))+\sum\limits_{i'':i\neq i',i,k}\bar{\rho}_{i''\to k}\end{array}\right], & \text{for } c_i\neq k\neq i\end{cases} \tag{29}$$

下面求解 $\tilde{\rho}_{i\to k}(c_i=k)=\rho_{i\to k}(c_i=k)-\bar{\rho}_{i\to k}$ 和 $\tilde{\alpha}_{i\leftarrow k}(c_i=k)=\alpha_{i\leftarrow k}(c_i=k)-\bar{\alpha}_{i\leftarrow k}$,以得到更简单的迭代公式:

$$\begin{aligned}\tilde{\rho}_{i\to k}(c_i=k)&=\rho_{i\to k}(c_i=k)-\bar{\rho}_{i\to k}\\ &=\rho_{i\to k}(k)-\max_{j:j\neq k}\rho_{i\to k}(j)\\ &=s(i,c_i)+s'(i,c_i')+\sum_{k':k'\neq k}\bar{\alpha}_{i\leftarrow k'}+[c_i\in neb(c_k')]\cdot\lambda_2\cdot I-\max_{j:j\neq k}\left[s(i,j)+s'(i,c_i')+\tilde{\alpha}_{i\leftarrow j}(j)+\sum_{k':k'\neq k}\bar{\alpha}_{i\leftarrow k'}\right]\end{aligned} \tag{30}$$

$$\begin{aligned}\tilde{\alpha}_{i\leftarrow k}(c_i=k)&=\alpha_{i\leftarrow k}(c_i=k)-\bar{\alpha}_{i\leftarrow k}\\ &=\begin{cases}[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\sum\limits_{i'':i''\neq i',k}\max(0,\tilde{\rho}_{i''\to k}(j'))+\\ \sum\limits_{i'':i''\neq i',k}\bar{\rho}_{i''\to k}-\sum\limits_{j':j'\neq k}\bar{\rho}_{j'\to k}, & \text{for } k=i\\ \tilde{\rho}_{k\to k}(k)+\bar{\rho}_{k\to k}+[k\in neb(c_k')]\cdot\lambda_2\cdot I+[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\\ \sum\limits_{i'':i\neq i',i,k}\max(0,\rho_{i''\to k}(j'))+\sum\limits_{i'':i\neq i',i,k}\bar{\rho}_{i''\to k}-\Omega\ , & \text{for } k\neq i\end{cases}\end{aligned} \tag{31}$$

这里的Ω可以表示为

$$\max\left[\begin{array}{l}\sum\limits_{j':j'\neq k}\bar{\rho}_{j'\to k}\\ \tilde{\rho}_{k\to k}(k)+\bar{\rho}_{k\to k}+[k\in neb(c_k')]\cdot\lambda_2\cdot I+\\ [k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\bar{\rho}_{i'\to k}+\lambda_2\cdot I)+\sum\limits_{i'':i\neq i',i,k}\max(0,\rho_{i''\to k}(j'))+\sum\limits_{i'':i\neq i',i,k}\bar{\rho}_{i''\to k}\end{array}\right].$$

从而,等式(31)可进一步化简为

$$\begin{cases}[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq k}}(\tilde{\rho}_{i'\to k}(k)+\lambda_2\cdot I)+\sum\limits_{i'':i''\neq i',k}\max(0,\tilde{\rho}_{i''\to k}(j')), & \text{for } k=i\\ \tilde{\rho}_{k\to k}(k)+[k\in neb(c_k')]\cdot\lambda_2\cdot I+[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\lambda_2\cdot I)+\sum\limits_{i'':i\neq i',i,k}\max(0,\rho_{i''\to k}(j'))- & \\ \max\left[0,\tilde{\rho}_{k\to k}(k)+[k\in neb(c_k')]\cdot\lambda_2\cdot I+[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval,\\ i'\neq i,k}}(\tilde{\rho}_{i'\to k}(k)+\lambda_2\cdot I)+\sum\limits_{i'':i\neq i',i,k}\max(0,\rho_{i''\to k}(j'))\right], & \\ \text{for } k\neq i\end{cases} \tag{32}$$

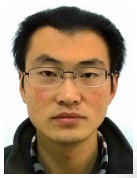
在更新过程中,当 $c_i\neq k$ 时,$\tilde{\rho}_{k\to i}(c_i)$ 和 $\tilde{\alpha}_{i\leftarrow k}(c_i)$ 并没有使用到(且注意到 $\tilde{\alpha}_{i\leftarrow k}(c_i\neq k)=0$),因此,传递的消息可以认为是 $r(i,k)=\tilde{\rho}_{k\to i}(k)$ 和 $a(i,k)=\tilde{\alpha}_{i\leftarrow k}(k)$:

$$r(i,k)=s(i,k)+s'(i,k)+[k\in neb(c_k')]\cdot\lambda_2\cdot I-\max_{j:j\neq k}[s(i,j)+s'(i,j)+a(i,j)] \tag{33}$$

$$a(i,k)=\begin{cases}[k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval\\ i'\neq k}}(r(i',k)+\lambda_2\cdot I)+\sum\limits_{i'':i''\neq i',k}\max(0,r(i'',k)), & k=i\\ \min\left[0,\ [k\in neb(c_k')]\cdot\sum\limits_{\substack{i':s(i',k)\leqslant maxval\\ i'\neq i}}(r(i',k)+\lambda_2\cdot I)+\sum\limits_{i'':i''\neq i'}\max(0,r(i'',k))\right], & k\neq i\end{cases} \tag{34}$$

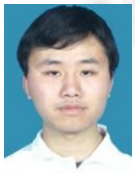
公式(34)就是文中给出的迭代公式.在迭代计算 $a(i,k)$更新时,min[0,·]的计算由如下等式得到:

$$x-\max(0,x)=\min(0,x).$$



杭文龙(1988－),男,江苏南通人,博士生,CCF 学生会员,主要研究领域为模式识别,数据挖掘.

刘解放(1982－),男,博士生,主要研究领域为模式识别,数据挖掘.



蒋亦樟(1988－),男,博士,讲师,CCF 专业会员,主要研究领域为模式识别,系统建模.

王士同(1964－),男,教授,博士生导师,主要研究领域为模式识别,人工智能.