文章编号:1672-3961(2013)05-0013-06 DOI:10.6040/j.issn.1672-3961.1.2013.195

基于 GraphLab 的分布式近邻传播聚类算法

陈文强[1],林琛[1,2],陈珂[3],陈锦秀[1],邹权[1,2*]
(1. 厦门大学信息科学与技术学院,福建 厦门 361005; 2. 厦门大学深圳研究院,广东 深圳 518057;
3. 广东石油化工学院计算机科学与技术系,广东 茂名 525000)

摘要:为有效实现海量数据的非线性聚类,提出基于 GraphLab 的分布式流式近邻传播算法——GStrAP(GraphLab based stream affinity propagation)。该算法将数据抽象为有向无环图模型,采用"Gather-Apply-Scatter"的模式完成数据同步和算法迭代。在人工合成流形数据 3D Clusters、Aggregation、Flame 和 Pathbased 数据集上分别采用不同数据规模以及与传统 K-means 的聚类性能做对比,实验表明:基于 GraphLab 的近邻传播算法对数据规模具有良好的拓展性,在保持算法聚类效果的同时,有效降低时间复杂度。

关键词:近邻传播聚类算法;分布式计算;GraphLab;聚类融合
中图分类号:TP301 **文献标志码:**A

Distributed affinity propagation clustering algorithm based on GraphLab

CHEN Wen-qiang[1], LIN Chen[1,2], CHEN Ke[3], CHEN Jin-xiu[1], ZOU Quan[1,2*]
(1. School of Information Science and Technology, Xiamen University, Xiamen 361005, China;
2. Shenzhen Research Institute, Xiamen University, Shenzhen 518057, China;
3. Department of Computer Science and Technology, Guangdong University of Petrochemical Technology, Maoming 525000, China)

Abstract: A distributed affinity propagation algorithm based on GraphLab was proposed, which was named GStrAP (Graphlab based stream affinity propagation). In GraphLab's DAG abstraction, the parallel computation was represented as a directed acyclic graph with data flowing along edges between vertices, and the "Gather-Apply-Scatter" paradigm was applied to complete data synchronization and algorithm's iteration. The experimental results on 3D Clusters, Aggregation, Flame and Pathbased datasets with different scale and the clustering performance were compared with K-means, which demonstrated that the proposed GStrAP could achieve high performance on both scalability and accuracy.
Key words: affinity propagation clustering algorithm; distributed computation; GraphLab; clustering ensemble

0 引言

聚类算法是数据挖掘、模式识别等研究方向的重要内容之一,在识别数据的内在结构方面具有极其重要的作用。然而随着信息技术的发展,互联网数据的规模和增长速度对存储和计算的能力提出了更高要求。为实现海量数据聚类,前人设计了并行化的聚类算法,如并行 K 均值聚类[1-2],分布式谱聚类算法[3-4]和基于MapReduce[5]的并行化近邻传播

收稿日期:2013-05-14 网络出版时间:2013-07-29 10:31
网络出版地址:http://www.cnki.net/kcms/detail/37.1391.T.20130729.1031.013.html
基金项目:国家自然科学基金资助项目(61102136,61001013);福建省自然科学基金资助项目(2011J05158,2010J01351);深圳市科技创新基础研究资助项目(JCYJ20120618155655087)
作者简介:陈文强(1988-),男,福建厦门人,硕士研究生,主要研究方向为可信信息系统研究. E-mail:chenwq@stu.xmu.edu.cn
* 通讯作者:邹权(1982-),男,黑龙江佳木斯人,讲师,硕士研究生导师,博士,主要研究方向为生物信息学与 Web 数据挖掘研究. E-mail:zouquan@xmu.edu.cn

算法[6]等。与 K 均值聚类相比较,近邻传播算法不需要预先设置聚类中心个数,适合处理非线性可分的数据[7]。

近邻传播(affinity propagation, AP)聚类[7-15]采用吸引度矩阵与归属度矩阵表示所有样本之间的关系,初始时认为所有样本都是潜在聚类中心。通过消息传递(信念传播)来更新上述两个矩阵,逐步确定质量高的聚类中心。在近邻传播聚类中,样本之间的距离不局限于三角不等式和距离对称等条件,可以采用任意距离度量方式。在 Frey 实现的近邻传播聚类中迭代更新吸引度矩阵和归属度矩阵的时间复杂度是 $O(N^2)$,对海量数据处理能力受限。

鲁伟明等人研究了基于 MapReduce 的分布式近邻传播聚类[6]。采用 SIMD 模型的 MapReduce 分布式计算框架在 Map 阶段集群的各台机器各自完成负载较重的计算过程,数据并行度高,适合完成类似矩阵运算、数据统计等数据独立性强的计算。而机器学习算法通常具有下面两个特点:数据依赖性强,运算过程各个机器之间要进行频繁的数据交换;流处理复杂,整个处理过程需要多次迭代,因此 MapReduce 的并行计算性能不高。本研究提出基于 GraphLab[16-18]的分布式近邻传播聚类算法——GStrAP,支持稠密数据稀疏化和计算并行的近邻传播聚类。虽然数据稀疏化可采用 t-近邻方法,但并不适用于流型数据,也难以确定近邻数。而采用近邻传播聚类不仅易于控制数据稀疏化效果,稀疏过程也易于并行。GStrAP 首先按照某种策略划分数据,得到规模相近的子集,然后在各个子集上进行消息传递,最后融合各子集结果进行下一次聚类的迭代。收敛后得到聚类代表,并为整个数据集指派聚类中心。

1 相关工作

1.1 GraphLab 计算模型

GraphLab 是近年来兴起的分布式机器学习框架,基于内存共享机制。支持稀疏的计算依赖、异步迭代计算等,解决了 MapReduce 不适应需要频繁数据交换的迭代机器学习算法,如近邻传播聚类算法的性能瓶颈。GraphLab 将数据抽象成 Graph 结构:$G=\{V,E,D\}$,其中 D 是用户自定义类型,顶点集 $\{v:v\in V\}$ 以及边集 $\{\mu\leftrightarrow v:\{\mu,v\}\in E\}$。GraphLab 以最小化集群计算节点之间的通信量和计算节点上的计算和存储均衡为原则对图模型表示的数据进行切分。例如,将与顶点 V 邻接的全部顶点划分为若干子集,对各个子集的计算分配到分布式集群上,各台机器上并行进行部分求和运算。通过集群中的 Master 节点和 Mirror 节点的通信完成最终的计算。其中,集群中的一台机器作为 Master 节点,其余机器上作为 Mirror。Master 作为所有 Mirror 的管理者,负责给 Mirror 安排具体计算任务,Mirror 作为节点在各台机器上的代理执行者,与 Master 数据的保持同步。

该模型将计算抽象为 3 个子过程:Gather、Apply和 Scatter。如图 1,在 Gather 过程,分布式计算的子集结果汇总到工作顶点 v,为下一步在 Apply 过程更新顶点 v 的顶点信息以及邻接节点的边信息做准备。

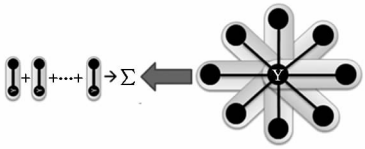

图1 Gather 阶段:收集当前节点的邻接节点信息

Fig. 1 Gather phase: accumulate information from the current vertex's neighborhoods

在 Apply 阶段,如图 2 所示:

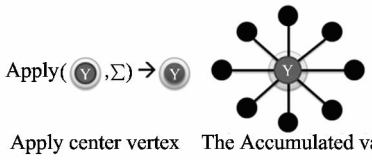

图2 Apply 阶段:用邻接节点的信息更新当前节点的信息

Fig. 2 Apply phase: apply the accumulated information from neighborhoods to update the current vertex

Mirror 将 Gather 阶段的计算结果发送给 Master,即 Gather 阶段计算完成之后,Master 进行汇总并更新 Master 节点的数据,如工作顶点 v 的顶点信息和边信息等。接着顶点更新后的信息同步到 Mirror 上。

最后,在 Scatter 阶段,如图 3 所示,更新当前工作节点的邻接节点的边上的数据,并通知对其有依赖的邻接顶点更新状态。完整 GAS 过程如图 4 所示。

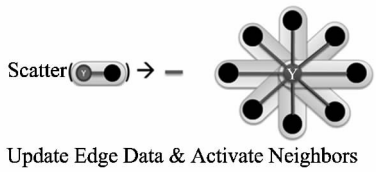

图3 Scatter 阶段:更新邻接节点的边上信息和节点自身的信息

Fig. 3 Scatter phase: update information of neighborhoods' edges and the current vertex's information

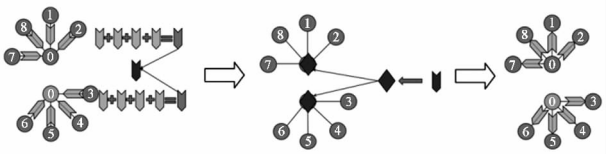

图 4　GraphLab 的 Gather-Apply-Scatter 模型

Fig. 4　The Gather-Apply-Scatter paradigm of GraphLab

该模型对分布式计算进行了高层次抽象,有助于高效快速实现迭代的机器学习算法,用户只需要考虑算法的实现逻辑,无需关心数据的图模型表示、集群节点之间的通信、一致性和容错性等细节问题。

1.2　近邻传播聚类算法

假设 $E=\{e_1,\cdots,e_N\}$ 为 N 个样本,$d(i,j)$ 表示样本 e_i 和 e_j 之间的距离或者差异,$\sigma(i)$ 表示样本 e_i 的代表点(examplar)的下标,则近邻传播聚类优化目标:

$$E[\sigma]=\sum_{i=1}^{N}S(e_i,e_{\sigma(i)})-\sum_{i=1}^{N}\chi_i[\delta],\quad(1)$$

其中,$S(e_i,e_{\delta(i)})=\begin{cases}-d(i,j), & \text{if } i\neq j;\\ -s^*, & \text{else。}\end{cases}$ 偏好系数 $s^*\geqslant 0$。$\chi_i[\delta]=\begin{cases}\infty, & \text{if } \delta(\delta(i))\neq\delta(i);\\ 0, & \text{else。}\end{cases}$,即如果样本 e_i 被其他样本选为代表点(examplar),那么 e_i 的代表点为自身,$\chi_i[\delta]=\infty$,否则为 0。

公式(1)通过消息传播(信念传播)算法求解,其中涉及两种信息参数:$r(i,k)$(responsibility)是样本 ε_i 指向样本 ε_k,代表 ε_k 积累的证据,用来表示 ε_k 适合作为 ε_i 的聚类代表点的程度。$a(i,k)$(availability)是从样本 ε_k 指向 ε_i 积累的证据,用来表示 ε_i 选择 ε_k 作为聚类代表点的合适程度。初始时,$r(i,k)$ 和 $a(i,k)$ 设置为 0,此后根据以下规则更新:

$$r(i,k)=S(e_i,e_k)-\max_{k',k'\neq k}\{a(i,k')+S(e_i+e_{k'})\},\quad(2)$$

$$r(k,k)=S(e_k,e_k)-\max_{k',k'\neq k}\{S(e_k+e_{k'})\},\quad(3)$$

$$a(k,k)=\sum_{i',i'\neq k}\max\{0,r(i',k)\},\quad(4)$$

$$a(i,k)=\min\{0,r(k,k)+\sum_{i',i'\neq i,k}\max\{0,r(i',k)\}\},\quad(5)$$

最后,样本 ε_i 的代表点 $\sigma(e_i)$ 的下标定义如下:

$$\sigma(i)=\arg\max\{r(i,k)+a(i,k),k=1\cdots N\},\quad(6)$$

算法迭代到指定最大迭代次数或者代表点在一定迭代次数后不变时停止。

2　基于 GraphLab 的分布式近邻传播聚类

根据上一节对近邻传播聚类算法思想的分析,基于 GraphLab 并行化更新公式(2)~(5)需要两次迭代:

(1) 更新 $r(i,k)$ 和 $r(k,k)$

(a) Gather

计算与顶点 Y 通过入边相连接的所有顶点之间 $a(i,k')+s(i,k')$ 最大的值,记为 max,以及第二大值,记为 second-max-value,如表 1 所示。

(b) Apply

根据公式(2),更新顶点 Y 的 $r(k,k)$ 值,如表 2 所示。

(c) Scatter

根据公式(3),更新与 Y 通过在入边上相邻的所有边的 $r(i,k)$ 值,如表 3 所示。

(2) 更新 $a(i,k)$ 和 $a(k,k)$

(a) Gather

对顶点 Y 通过出边相连的所有顶点求 $\sum_{i',i'\neq k}\max\{0,r(i',k)\}$,记为 sum,如表 1。

(b) Apply

根据公式(5),更新顶点 Y 的 $a(k,k)$ 值,如表 2。

(c) Scatter

根据公式(4),更新与 Y 通过在出边上相邻的所有边的 $a(i,k)$ 值,如表 3 所示。

表 1　基于 GraphLab 的近邻传播算法的 Gather 阶段

Table 1　The Gather phase of GStrAP

```
Gather(vertex, edge, damping)
1.  input: a center vertex and one subset of the edges connected to it
2.  output: an intermediate result of max and second-max-value
3.  Begin
4.  max = 0, second-max-value = 0, sum = 0
5.  if update r(k,k) and r(i,k)
        if (edge.r + edge.a > max)
6.          second-max-value = max;
            max = edge.r + edge.a;
        If (edge.r + edge.a < = max and edge.r +
7.  edge.a > = second-max-value)
            second-max-value = edge.r + edge.a;
8.  if update a(k,k) and a(i,k)
9.      sum + = max(edge.r, 0.0)
```

表2 基于GraphLab的近邻传播算法的Apply阶段
Table 2 The Apply phase of GStrAP

```
Apply(vertex, max, second-max-value, sum, damping)
1.  input: a center vertex, max, second-max-value,
      and the sum calculated by Gather phase
2.  output: the updated vertex
3.  Begin
4.  old_r = vertex. r, old_a = vertex. a
5.  if update r(k,k)
        if (max is from vertex itself)
6.          vertex. r = vertex. s + second-max-value;
            max = edge. r + edge. a;
7.      If (max is not from vertex itself)
            vecrtex. r = vertex. s + max;
8.      vertex. r = (1 - damping) * vertex. r +
    damping * old_r;
8.  if update a(k,k)
9.      vertex. a = sum;
10.     vertex. a = (1 - damping) * vertex. a +
    damping * old_a;
```

表3 基于GraphLab的近邻传播算法的Scatter阶段
Table 3 The Scatter phase of GStrAP

```
Scatter(vertex, edge, damping)
1.  input: a center vertex and one subset of the
      edges connected to it
2.  output: the updated edge
3.  Begin
4.  max = 0, second-max-value = 0, sum = 0, old_r =
    edge. r, old. a = edge. a;
5.  if update r(i,k)
6.      if (vertex. r + vertex. a > max)
            edge. r = edge. s - max;
7.      If (vertex. r + vertex. a < = max)
            edge. r = edge. s - second-max-value;
8.      edge. r = (1 - damping) * edge. r + damping
    * old. r;
8.  if update a(i,k)
9.      edge. a = min(0. 0, vertex. r - max(0. 0,
    edge. r));
        edge. a = (1 - damping) * edge. a +
    damping * old. a;
```

3 实验结果与分析

3.1 实验的数据集

本研究采用流形学习工具MANI[19]合成的数据3D Clusters、Aggregation[20]、Flame[21]以及Pathbased[22]作为数据集。其中3D Clusters数据由N个点构成互不重叠M个聚类,M个聚类之间用直线相连。以$N=188$,$M=3$为例生成的数据如图5所示。

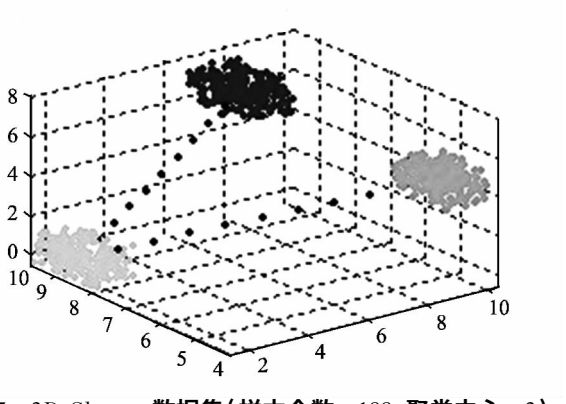

图5 3D Clusters数据集(样本个数=188,聚类中心=3)
Fig. 5 3D Clusters dataset(instances=188, clusters=3)

Aggregation、Flam和Pathbased数据集的信息如表4所示。

表4 Aggregation、Flame和Pathbased数据集的信息
Table 4 The informations of Aggregation, Flame and Pathbased datasets

数据集	样本数	属性数	聚类中心数
Aggregation	788	2	7
Flame	240	2	2
Pathbased	300	2	3

采用的3D Clusters数据集的规模分别为1 000,2 000,4 000和8 000。

3.2 聚类的评价指标

采用同质性(homogeneity),完整性(completeness),V-measure[23],Adjusted Rand Index[24]以及NMI(normalized mutual information)[25-26]来度量GStrAP的聚类效果。

(1) Homogeneity

聚类算法的每个聚类中心分别只包含同一种类别的样本,则称聚类结果满足同质性。Homogeneity取值在[0,1]之间,值越高表示聚类效果越好。

$$h=\begin{cases}1, \text{ if } H(C,K)=0;\\ 1-\dfrac{H(C\mid K)}{H(C)}, \text{ else。}\end{cases} \tag{7}$$

其中$H(C)=-\sum\limits_{c=1}^{|C|}\dfrac{\sum\limits_{k=1}^{|K|}a_{ck}}{n}\log\dfrac{\sum\limits_{k=1}^{|K|}a_{ck}}{n}$,

$H(C\mid K)=-\sum\limits_{k=1}^{|K|}\sum\limits_{c=1}^{|C|}\dfrac{a_{ck}}{N}\log\dfrac{a_{ck}}{\sum\limits_{c=1}^{|C|}a_{ck}}$。

(2) Completeness

聚类算法的每个聚类中心包含所有属于同一个类别的样本,则称聚类结果满足完整性。Completeness取值也在[0,1]之间,值越高说明聚类效果越好。

$$c=\begin{cases}1, \text{ if } H(K,C)=0;\\ 1-\dfrac{H(K|C)}{H(K)}, \text{ else。}\end{cases} \tag{8}$$

其中 $H(K)=-\sum_{k=1}^{|K|}\dfrac{\sum_{c=1}^{|C|}a_{ck}}{n}\log\dfrac{\sum_{c=1}^{|C|}a_{ck}}{n}$,

$H(K|C)=-\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\dfrac{a_{ck}}{N}\log\dfrac{a_{ck}}{\sum_{k=1}^{|K|}a_{ck}}$。

(3) V-measure

V-measure 是 Homogeneity 和 Completeness 的调和平均数,如下定义:

$$v=2*\frac{(\text{homogeneity}*\text{completeness})}{(\text{homogeneity}+\text{completeness})}。 \tag{9}$$

(4) Adjusted Rand Index(ARI)

Rand Index(RI)常用在衡量两个聚类之间的相似度,Adjusted Rand Index(ARI)是 Rand Index 的改进。

$$\text{ARI}=\frac{(\text{RI}-\text{expected\_RI})}{(\max(\text{RI})-\text{expected\_RI})}。 \tag{10}$$

(5) NMI

对随机变量 X 和 Y,NMI 的定义为 $\text{NMI}(X,Y)=\dfrac{I(X,Y)}{\sqrt{H(X)H(Y)}}$,其中 $I(X,Y)$ 是 X 和 Y 之间的互信息,$H(X)$,$H(Y)$ 分别是 X,Y 的熵。对给定聚类结果,$\text{NMI}(X,Y)$ 的计算如下:

$$\text{NMI}=\frac{\sum_{k=1}^{C}\sum_{m=1}^{C}n_{k,m}\log\left(\dfrac{n\cdot n_{k,m}}{n_k\cdot\hat{n}_m}\right)}{\sqrt{\left(\sum_{k=1}^{C}n_k\log\left(\dfrac{n_k}{n}\right)\right)\left(\sum_{m=1}^{C}\hat{n}_m\log\left(\dfrac{n_m}{n}\right)\right)}}, \tag{11}$$

其中 n_k 表示聚类结果中第 k 类,C_k($1\leqslant k\leqslant C$)的样本个数,$\hat{n}_m$ 表示真实数据的第 m 类,L_m($1,\leqslant m\leqslant C$)的个数,$n(k,m)$ 为 C_k 和 L_m 交集的个数。$\text{NMI}(X,Y)$ 值越大,表明聚类效果越好。

3.3　GStrAP 的性能表现

对上述数据集分别使用原始近邻传播聚类和本研究实现的 GStrAP 算法进行聚类,采用欧式距离作为相似性度量,最大迭代次数设为 200。

图 6 记录 AP(Serial AP)和 GStrAP(Distributed AP)在不同规模数据集上的运行时间,横坐标为样本个数,纵坐标为聚类时间,单位 s。原始近邻传播算法和 GStrAP 均运行于 3.10 GHz 的 4 核处理器上,内存为 8 GB。当样本规模较小时,GStrAP 与原始近邻传播运行时间差不多。当样本规模较大时,GStrAP 运行时间为原始近邻传播的 $2/C$,C 为分布式集群中工作机子的个数。可以看出,分布式聚类有效缩短聚类时间。

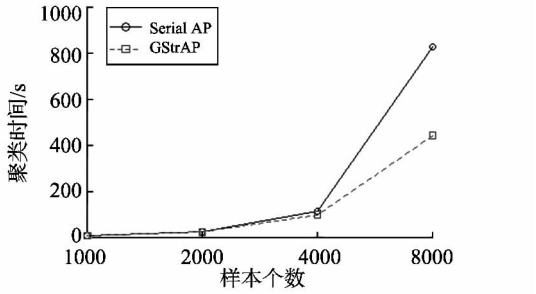

图 6　不同规模 3D Clusters 数据集上 AP 和 GStrAP 的时间对比

Fig. 6　Processing time comparison between AP and GStrAP on 3D Clusters with different size

表 5 记录了在不同规模 3D Clusters 数据集上,原始近邻传播聚类和 GStrAP 进行 200 次迭代的结果。

表 5　不同规模数据集上 AP 聚类和 GStrAP 聚类的时间对比

Table 5　Processing time comparison between AP and GStrAP on 3D Clusters with different size

Data scale	Vertex number	Edges number	AP runtime/s	GStrAP runtime/s
1 000	1 000	1 000 000	8.163	6.903
2 000	2 000	4 000 000	25.562	23.200
4 000	4 000	16 000 000	114.983	98.869
8 000	8 000	64 000 000	830.162	444.060

为检验 GStrAP 的聚类性能,在 Aggregation、Flame 和 Pathbased 数据集上分别使用 GStrAP 和 K-means进行对比实验,聚类性能信息如表 6 所示。

表 6　GStrAP 和 K-means 在不同数据集上的聚类性能对比

Table 6　Clustering performance comparison between GStrAP and K-means on different dataset

Dataset	Algorithm	K	Homogeneity	Completeness	V-measure	ARI	NMI
Aggregation	GStrAP	7	0.879	**0.799**	**0.837**	0.703	**0.837**
	K-means	7	0.879	0.797	0.836	**0.737**	0.836
Flame	GStrAP	2	**0.482**	**0.456**	**0.468**	0.488	**0.468**
	K-means	2	0.475	0.449	0.462	**0.500**	0.462
Pathbased	GStrAP	3	**0.509**	0.582	**0.543**	**0.458**	**0.543**
	K-means	3	0.401	**0.634**	0.491	0.399	0.491

4 结语

本研究提出了基于 GraphLab 的分布式流式近邻传播算法——GStrAP,克服了近邻传播算法在 SIMD 模型并行性能低的缺点,为海量数据的聚类提供了有效的解决办法。实验表明 GStrAP 对数据规模具有良好的拓展性,在保持算法聚类效果的同时,能够有效降低聚类的时间复杂度。实验所采用数据集的相似度矩阵均是稠密的,对于稠密相似度矩阵,GStrAP 的时间复杂度是 $O(N^2/C)$,C 为集群中工作机个数。对稀疏相似度矩阵进一步研究,设计空间和时间复杂度更低的分布式近邻传播聚类是作者今后要研究的一方面。

参考文献:

[1] WEIZHONG Z, HUIFANG M, QING H. Parallel *K*-means clustering based on MapReduce [C] // Proceedings of the 1st International Conference on Cloud Computing. Berlin, Germany: CloudCom, 2009: 674-679.

[2] ATTILA G. Data decomposition for parallel *K*-means clustering [C] // Proceedings of the 5th International Conference on Parallel Processing and Applied Mathematics. Czestochowa, Poland: PPAM, 2003: 241-248.

[3] WENYEN C, YANGQIU S, HONGJIE B, et al. Parallel spectral clustering in distributed systems [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, 33(3):568-586.

[4] YANGQIU S, WENYEN C, HONGJIE B, et al. Parallel spectral clustering [J]. ECML PKDD, 2008, 374-389.

[5] DEAN J, GHEMAWAT S. MapReduce: simplified data processing on large clusters [J]. Communications of the ACM, 2008, 51(1):107-113.

[6] 鲁伟明,杜晨阳,魏宝刚,等. 基于 MapReduce 的分布式近邻传播聚类算法 [J]. 计算机研究与发展, 2012, 49(8):164-174.
LU Weiming, DU Chenyang, WEI Baogang, et al. Distributed affinity propagation clustering based on MapReduce [J]. Journal of Computer Research and Development, 2012, 49(8):164-174.

[7] TAPAS K, DAVID M, NATHAN S, et al. An efficient *K*-means clustering algorithm: analysis and implementation [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(7):881-892.

[8] BRENDAN J F, DUECK D. Clustering by passing messages between data points [J]. Science, 2007, 315(5814):72-976.

[9] KAIJUN W, JUNYING Z, DAN L, et al. Adaptive affinity propagation clustering [J]. Acta Automatica Sinica, 2007, 33(12):1242-1246.

[10] XIANGLIANG Z, CYRIL F, MICHELE S. Data streaming with affinity propagation [C] // Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases. Berlin, Germany: ECML PKDD, 2008: 628-643.

[11] DELBERT D, BRENDAN J F. Non-metric affinity propagation for unsupervised image categorization [C] // Proceedings of the 11th International Conference on Computer Vision. Rio de Janeiro, Brazil: ICCV, 2007: 1-8.

[12] MICHELE L, SUMEDHA M W. Clustering by soft-constraint affinity propagation: applications to gene-expression data [J]. Bioinformatics, 2007, 23(20): 2708-2715.

[13] BRENDAN J F, DELBERT D. Mixture modeling by affinity propagation [C] // Proceedings of the 19th Annual Conference on Neural Information Processing Systems. Vancouver, Canada: Elsevier, 2005: 379-386.

[14] ZHIWU L, HORACE, H S. Constrained spectral clustering via exhaustive and efficient constraint propagation [C] // Proceedings of the 11th European Conference on Computer vision. Crete, Greece: ECCV, 2010: 1-14.

[15] INMAR E G, BERNDAN J F. A binary variable model for affinity propagation [J]. Neural Computation, 2009, 21(6):1589-1600.

[16] YUCHENG L, BICKSON D, GONZALEZ J, et al. Distributed GraphLab: a framework for machine learning and data mining in the cloud [J]. Proceedings of the VLDB Endowment, 2012, 5(8):716-727.

[17] JOSEPH E G, YUCHENG L, HAIJIE G, et al. PowerGraph: distributed graph-parallel computation on natural graphs [C] // Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation. San Diego, USA: OSDI, 2012: 17-30.

[18] YUCHENG L, JOSEPH E G, AAPO K, et al. GraphLab: a new framework for parallel machine learning [C] // Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence. Catalina Island, California: UAI, 2010: 340-349.

[19] TODD W. Mani fold learning matlab demo [CP/OL]. [2011-08-14]. http://www.math.ucla.edu/~wittman/mani/.

[20] GIONIS A, MANNILA P T. Clustering aggregation [J]. ACM Transactions on Knowledge Discovery from Data (TKDD), 2007, 1(1):1-30.

(下转第23页)

poly-connection [J]. Journal of the China railway Society, 2012, 34(2):52-57.

[8] EMIN YÜKSEL M. Edge detection in noisy images by neu-ro-fuzzy processing [J]. International Journal of Electronics and Communications, 2007, 61:82-89.

[9] CANNY J F. A computational approach to edge detection [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986, 8(6):679-698.

[10] WAN Shuai, YANG Fuzheng, HE Mingyi. Gradient — th-reshold edge detection based on perceptually adaptive thre-shold selection [C] //Proceedings of the 3rd Conference on Industrial Electronics and Applications. Singapore: IEEE, 2008:999-1002.

[11] 徐平,邵定宏,魏楹. 最佳阀值分割和轮廓提取技术及其应用 [J]. 计算机工程与设计,2009,30(2):437-439.
XU Ping, SHAO Dinghong, WEI Ying. Optical threshold segmentation and contour extraction technology and its application [J]. Computer Engineering and Design, 2009, 30(2):437-439.

[12] 何文浩,原魁,邹伟. 自适应阀值的边缘检测算法及其硬件实现 [J]. 系统工程与电子技术,2009,31(1):233-237.
HE Wenhao, YUAN Kui, ZOU Wei. Self-adaptive threshold edge detection and its implementation in hardware [J]. Systems Engineering and Electronics, 2009, 31(1):233-237.

[13] MEDINA R, MUÑOZ R, YEGUAS E, et al. A novel method to look for the hysteresis thresholds for the Canny edge detector [J]. Pattern Recognition, 2011, 44(6):1201-1211.

[14] 陈强,黄声享,王韦. 小波去噪效果评价的另一指标 [J]. 测绘信息与工程,2008,33(5):13-14.
CHEN Qiang, HUANG Shengxiang, WANG Wei. An eva-luation indicator of wavelet denoising [J]. Journal of Geomatics, 2008, 33(5):13-14.

[15] CHEN Qiang, SUN Quansen, PHENG Annheng, et al. A double threshold image segmentation binarization method based on edge detector [J]. Pattern Recognition, 2008, 41(4):1254-1267.

[16] 姚伟,孙即祥. 图像去噪中的纹理保护方法研究 [J]. 中国图象图形学报,2010,15(5):723-728.
YAO Wei, SUN Jixiang. Studies on texture preserving image denoising methods [J]. Journal of Image and Graphics, 2010, 15(5):723-728.

[17] MYAKININ O O, KOMILIN D V, BRATCHENKO I A, et al. Noise reduction method for OCT images based on empirical mode decomposition [J]. Journal of Innovative Optical Health Sciences, 2013, 6(2):135009-1-135009-5.

[18] 胡爱军,向玲,唐贵基,等. 基于数学形态变换的转子故障特征提取方法 [J]. 机械工程学报,2011,47(23):92-95.
HU Aijun, XIANG Ling, TANG Guiji, et al. Fault feature extracting method of rotating machinery based on mathematical morphology [J]. Chinese Journal of Mechanical Engineering, 2011, 47(23):92-95.

[19] FABIO Boschetti. Improved edge detection and noise removal in gravity maps via the use of gravity gradients [J]. Journal of Applied Geophysics, 2005, 57(3):213-225.

[20] LI Jing, HUANG Peikang, WANG Xiaohu, et al. Image edge detection based on beamlet transform [J]. Journal of Systems Engineering and Electronics, 2009, 1:1-5.

[21] ZHANG Ying. Face and lip tracking for person identification [J]. China Communications, 2010, 7(6):141-144.

(编辑:陈燕)

(上接第 18 页)

[21] FU L, MEDICO E. FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data [J]. BMC bioinformatics, 2007, 8(1):3.

[22] CHANG H, YEUNG D Y. Robust path-based spectral clustering [J]. Pattern Recognition, 2008, 41(1):191-203.

[23] ANDREW R, JULIA H. V-Measure: a conditional entropy-based external cluster evaluation measure [C] // Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Prague, Czech Republic: EMNLP-CoNLL, 2007: 410-420.

[24] HUBERT L, ARABIE P. Comparing partitions [J]. Journal of Classification, 2008, 2(1):193-218.

[25] STREHL A, GHOSH J. Cluster ensembles — a knowledge reuse framework for combining multiple partitions [J]. Journal of Machine Learning Research, 2002, 3:583-617.

[26] NGUYEN X V, JULIEN E, JAMES B. Information theoretic measures for clusterings comparison: is a correction for chance necessary? [C] // Proceedings of the 26th Annual International Conference on Machine Learning. Montreal, Canada: ICML, 2009: 1073-1080.

(编辑:胡春霞)