文章编号: 1001-0920 (2012) 04-0542-05

基于 Squeezer 算法的文本数据流聚类

尤薇佳[1], 刘 鲁[1], 刘 丹[1], 李 明[2]

(1. 北京航空航天大学 经济管理学院，北京 100191；2. 中国石油大学 工商管理学院，北京 102249)

摘 要: 为解决数据流聚类中的“链式数据”问题以及文本数据流存在的高维、稀疏、多主题问题, 以 Squeezer 聚类算法为基础, 重新定义了聚类过程中类的质心、半径和判别距离. 提出了一种改进算法, 通过加入数据预处理环节来提高聚类精度, 通过投影聚类提高聚类效率并为簇赋予语义. 最后通过在互联网新闻语料的聚类实验, 表明了所提出的算法能够以较小的速度代价换来聚类效果的大幅提升, 性能显著优于 Squeezer 算法.

关键词: 文本数据流；Squeezer 算法；投影聚类

中图分类号: TP311 **文献标识码:** A

Text stream clustering based on Squeezer algorithm

YOU Wei-jia[1], *LIU Lu*[1], *LIU Dan*[1], *LI Ming*[2]

(1. School of Economics and Management，Beihang University，Beijing 100191，China；2. School of Business Administration，China University of Petroleum，Beijing 102249，China. Correspondent：YOU Wei-jia，E-mail: weijiawx@gmail.com)

Abstract: To solve the problems of “chain data” and “high-dimension, multi-topic, large-scale text stream” in data stream clustering, a modified Squeezer clustering algorithm is proposed, which combines the idea of projected clustering and redefines the class centroid, radius, and judging distance. The preprocessing stage and the projected clustering stage are introduced to improve the performance significantly and attach the semantic to the clusters for better understanding respectively. The experiment on the Internet corpus shows that the cluster result is significantly improved at a small cost of speed decrease and the performance of the proposed algorithm is better than that of Squeezer algorithm.

Key words: text stream；Squeezer algorithm；projected clustering

1 引 言

随着 Web 2.0 的飞速发展, 互联网上的信息飞速膨胀, 每天新生成的页面数以千万计. 对网页内容聚类, 发现网民关注的热点并了解热点发展趋势, 对于整合网络信息和了解网络民意具有十分重要的意义. 但是, 以网页为表现形式的文档数量急剧增长, 内容随时间不断变化且趋近无限, 是一种典型的数据流场景, 传统的聚类算法并不适用[1].

文献 [2] 提出的 Squeezer 算法是一种主要应用于大规模数据集的聚类算法, 也常用于数据流问题. Squeezer 算法不需要先验知识, 基于分类属性数据和数值属性数据的最大相似度或最小距离 (差异) 原则, 只扫描数据集一遍便可将数据集分割成几个不同的超球体实现聚类, 且具有良好的稳定性. 之后有学者在此基础上进行改进, 提出了 ID-Squeezer 算法[3], 引入半径的概念以消除链式效应, 基于用区间数据来保存聚类结果的思想对一定时间窗口的数据流进行聚类, 当新的数据流到达时, 在一定的阈值范围内动态调整先前聚类后所得区间数据的上下限, 并存储数据所属类的标示. 但是, Squeezer 算法和 ID-Squeezer 在文本数据流聚类领域仍存在优化的空间: 首先, 这两种算法直接从第 1 个点开始聚类, 如果前几个点是数据集的奇异点, 则会对最后的聚类结果产生较大的影响; 其次, 文本向量的维度通常都非常高, 而这两种算法均未对高维空间聚类进行优化, 性能尚有提升的空间; 再次, 这两种算法虽然不需确定聚类结果的类别数, 但需要设定距离阈值和半径阈值, 这些阈值的设置通常依赖于经验, 尚有待改进.

收稿日期: 2010-10-26；**修回日期:** 2011-01-17.

基金项目: 国家自然科学基金项目(90924020)；教育部博士点基金项目(200800060005)；阿里巴巴青年学者支持计划项目(活水计划Ali-2010-B-6).

作者简介: 尤薇佳(1981–), 女, 博士生, 从事数据挖掘、电子商务的研究；刘鲁(1947–), 女, 教授, 博士生导师, 从事电子商务、知识管理等研究.

本文针对 Squeezer 算法存在的缺陷, 以 Squeezer 聚类算法为基础, 引入投影聚类算法的思想重新定义了类的质心、半径, 引入判别距离的概念较好地解决了文本向量的高维、稀疏、多主题问题. 同时通过开始时间窗数据预处理提高了聚类精度, 通过维度投影降低计算复杂度, 并为聚类结果加上了可理解的语义标签.

2　Squeezer 投影聚类算法

2.1　相关定义

定义高维空间上的数据点集 $X=\{x_1,x_2,\cdots,x_m\}$, 维度空间 $\mathrm{Dim}=\{d_1,d_2,\cdots,d_l\}$, 共 l 维.

定义 1　定义二元组 $\mathrm{CS}_i=(C_i,D_i)$为一个簇, $C_i=\{x_1,x_2,\cdots,x_t\}$ 代表簇中的全部数据点, D_i 为簇所关联的维度集合, C_i 中每一点都与 D_i 中的维度相关.

投影聚类是指将 X 中的所有数据点划分为 k 个部分 (簇): $\{\mathrm{CS}_1,\mathrm{CS}_2,\cdots,\mathrm{CS}_k\}$, k 为聚出的簇的数目, 不同簇的维度集的维数可能不同[2].

定义 2　在维度集 $D(D\subseteq\mathrm{Dim})$ 构成的子空间中, 文本向量空间中的两个点, 点 $x_1=\{x_1^1,x_1^2,\cdots,x_1^{l_d}\}$ 与点 $x_2=\{x_2^1,x_2^2,\cdots,x_2^{l_d}\}$ 的距离为

$$d_D(x_1,x_2)=\frac{\sum\limits_{i\in D}|x_1^i-x_2^i|}{|D|},$$

这是曼哈顿距离的一个变形[4].

定义 3　簇的质心是簇中各维上所有点的算术平均值组成的向量, 记作 $\overline{x}_c$, 有 $\overline{x}_c=\sum\limits_{i=1}^{t}\dfrac{x_i}{t}$.

定义 4　维的半径是维上全部点到质心距离的平均值, 即簇 C 中维度 d 的半径记作 r_c^d, 有

$$r_c^d=\sum_{i=1}^{t}\frac{d(\overline{x}_c,x_i)}{t}.$$

定义 5　维的标准化半径是对簇中各维的维度半径标准化之后的结果. 簇上维度 d 的标准化半径记作 zr_c^d, 有

$$zr_c^d=\frac{r_c^d-r_c}{\sigma_{r_c}}.$$

其中: 簇中各维半径的均值记作 $\overline{r}_c$, 有

$$\overline{r}_c=\sum_{d\in D}\frac{r_c^d}{|D|}.$$

各维半径的标准差记作 σ_{r_c}, 有

$$\sigma_{r_c}=\sqrt{\frac{\sum\limits_{d\in D}(r_c^d-\overline{r}_c)^2}{|D|-1}}.$$

定义 6　点到簇的投影距离是指在簇指定的维度上点到簇的质心的距离. 点 x_1 到簇 C 的投影距离记作 $d_{xi,c}$, 有 $d_{x_i,c}=d_D(x_1,\overline{x}_c)$, 其中簇的维度集 D 表示点的维度.

定义 7　簇的判别距离是指决定一个点能否划分到指定簇之中的阈值, 簇 C 的判别距离记作 jd_c, 有

$$jd_c=\delta\sqrt{\frac{\sum\limits_{D}\dfrac{\sum\limits_{k=1}^{t}d_D(x_k,\overline{x}_c^2)}{t-1}}{|D|}}.$$

其中: δ 为控制系数, 由用户指定; t 为簇中数据点的数量; $|D|$ 为该维度集的维数. 规定

$$\begin{cases}x_i\notin C,\ d_{x_i,c}\geqslant jd_c;\\ x_i\in C,\ d_{x_i,c}<jd_c.\end{cases}$$

定义 8　聚类结果的总维数是指聚类过程中和最终聚类得到的结果中, 各簇所选择的全部维度的数量用 $l_{\max}$ 表示.

2.2　算法描述

对于文本流, 首先对其进行文本特征提取, 即将字符串转变为文本向量, 将主题词映射到相应维度. 在数据流环境下, 因为历史信息不能完整保存, 需定义一个簇结构来保存簇中历史数据的概要信息以计算新的质心、半径以及判别距离等参数.

在聚类环节, 基于 Aggrewal[5]提出的投影聚类算法 (projected clustering) 的思想, 本文提出的文本流投影聚类算法包含两个阶段, 即初始化阶段和扫描分拣阶段. 初始化阶段使用传统的 k-means 方法对随机选择的部分数据进行全维度聚类, 获取初始簇并计算其质心、半径、维度集等统计特征; 扫描分拣阶段是在初始簇的基础上采用基于 Squeezer 算法的文本流投影聚类算法对数据点进行单遍扫描分拣, 为所有数据点分配最合适的簇, 若没有合适的簇, 则形成新簇, 直到所有点都计算完成.

2.3　算法步骤

2.3.1　生成文档向量

定义文本数据流

$$\mathrm{Doc}=\{\mathrm{doc}_1,\mathrm{doc}_2,\cdots,\mathrm{doc}_t,\cdots\},$$

其中 doc_t 为 t 时刻到达的文档数据. 中文词库中全部词的集合为 $W=\{w_1,w_2,\cdots,w_t,\cdots\}$, 对应于全维度空间. 对 doc_t 进行特征提取之后, 得到该文档的特征词向量 $X_t=\{w_t^1,w_t^2,\cdots,w_t^d\}$, $w_t^i\in W$. 其中: d 为所有不重复的特征词的数量, 向量第 i 维元素 w_t^i 的值代表该词的特征值.

将文本字符串转化为文本特征词向量, 是将每一文档都映射为词向量形成的向量空间中的一个点. 这种向量空间模型的表示方法最大的优点在于将非结构化和半结构化的文本表示为向量形式, 使得各种数学计算成为可能. 这里, 为了最大程度地区分不同文档, 需要构造一个评价函数来表示词在文中的特征值,

计算公式如下[6]:

$$w_t^i(\mathrm{doc}_t) = tw_i \times \mathrm{posw} \times \sum_j \left(dw_j \times \frac{1+\ln(1+\ln(tf_{ij}))}{(1-s)+s\dfrac{dl_j}{\mathrm{avdl}}} \times \ln\frac{l+1}{df_i} \right).$$

其中: tw_i 为词 t_i 的语义权重; posw 为词 t_i 的词性权重, 例如虚词 posw 直接设为 0; dw_j 为该文档的重要程度, 可根据文档的来源、时序等信息决定网页文档的权重, 简化处理时可全部取 1; dl_j 为该文档 doc_t 的总词数; avdl 为所有文档总长度的平均值, l 为文档的数量; tf_{ij} 为词的频度, 即词 t_i 在文档 doc_t 中出现的次数; s 为一个经验参数, 通常取 0.2[7].

2.3.2 进行文档流聚类

1) 初始化阶段

ID-Squeezer 算法和 Squeezer 算法都是随机选取一个点作为聚类起始点, 如果第 1 个起始点是奇异点, 且已作为簇中一员, 则簇中不断引入新的点将使质心偏移, 导致聚出的类的范围扩大, 即使利用 ID-Squeezer 算法引入的半径阈值概念, 也会因为簇中已存在奇异点, 同样导致半径扩大. 另外, 这两种算法都没有反复迭代的过程, 把点一次性划分到簇, 缺乏退出机制, 即使该奇异点实际上已经远离质心, 也缺乏有效手段将该点排除出聚类结果, 导致聚类精度降低.

本算法引入了系统初始化步骤, 即随机选取一部分节点作为初始节点集, 对这个节点集进行初步迭代聚类, 获得初步质心, 以有效地降低随机质心对聚类结果的影响. 在数据流环境下, 初始节点集可以设定为早期到达的一批数据点. 通过设定初始聚类数 k, 采用经典的 k-means 算法将初始到达的一批点聚成 $\mathrm{CS}_{\mathrm{initial}}=\{\mathrm{CS}_1, \mathrm{CS}_2, \cdots, \mathrm{CS}_k\}$.

2) 扫描分拣阶段

本步骤引入了判别距离的概念, 以取代原算法的距离阈值和半径阈值, 同时引入投影聚类的思想对高维数据在最优维度集的子空间进行降维, 以提高算法的性能. 在初始化阶段得到的初步簇基础上, 计算后续各数据点所属簇, 算法如下:

输入: 剩余数据集 $X=\{X_{m+1}, \cdots, X_n\}$, 初始簇集合 $\mathrm{CS}_{\mathrm{initial}}=\{\mathrm{CS}_1, \mathrm{CS}_2, \cdots, \mathrm{CS}_k\}$;

输出: 聚类结果 C.

算法步骤如下:

Step 1: 检查 X 是否为空, 若为空, 则转 Step 10, 算法结束.

Step 2: 从剩余数据点 X 中读入新的数据 X_t.

Step 3: 针对现有各簇, 在 X_t 到达之后, 计算最优投影维度集合, 重新分配各簇 CS_i 的维度集 D_i. 最优投影维度是最能反应聚类特征的维度, 即簇中该维度上的数据是最密集的. 而维上数据的密集程度, 可以利用将各维半径进行标准化之后的标准化半径来衡量 (参见定义 5), 维度标准化半径

$$zr_c^d = \frac{r_c^d - \overline{r}_c}{\sigma_{r_c}}.$$

其中: 各维半径的方差为 σ_{r_c}, 有

$$\sigma_{r_c} = \sqrt{\frac{\sum\limits_{d\in D}(r_c^d - \overline{r}_c)^2}{|D|-1}}.$$

$\overline{r}_c$ 为簇中各维半径的均值, 有

$$r_c^d = \sum_{i=1}^{t} \frac{d(\overline{x}_c, x_i)}{t}.$$

半径为

$$r_c^d = \frac{\sum\limits_{i=1}^{t} d(\overline{x}_c, x_i)}{t}.$$

这里, 可通过基于网格的概要数据提取方法来估算半径, 将维度分段, 切分成 n 个子区间, 增量计算区间 s_i 数据点的均值 $\overline{x}_{s_i}$ 和数据点的数量 c_i, 以均值点作为数据点的概要信息进行求半径的运算. 总数据点的数量为 t, 有

$$r_c^d \approx \frac{\sum\limits_{i=1}^{n} c_i \times |\overline{x}_i - \overline{x}_{s_i}|}{t}.$$

然后从小到大选取非零的前 $l_{\max}$ 个维度, 并将选出的维度集分配到各簇.

Step 4: 基于各簇的维度集, 遍历计算 X_t 与现有簇之间的投影距离 ds, 并找出投影距离最小的簇 $\mathrm{CS}_{\mathrm{index}}$ 以及最近投影距离 ds_{index}, 参见定义 6.

Step 5: 计算 $\mathrm{CS}_{\mathrm{index}}$ 的判别距离 jd_{index}. 由定义 7, 有

$$jd_c = \delta\sqrt{\frac{\sum\limits_{D}\dfrac{\sum\limits_{k=1}^{t} d_D(x_k, \overline{x}_c^2)}{t-1}}{|D|}}.$$

因为 δ 和 N 为常量, 故有

$$\sum_{x_k\in C} d_D(x_k, \overline{x}_c)^2 = \sum_{k=1}^{t}(x_k^2 - 2\times x_k \times \overline{x}_c + \overline{x}_c^2) = \sum_{k=1}^{t} x_k^2 - \frac{\left(\sum\limits_{k=1}^{t} x_k\right)^2}{t}.$$

由推导可知, 只要记录 $\sum x_k^2$ 和 $\sum x_k$ 便可完成计算, 并不需要记录全部的历史数据信息. 对于簇 CS_i 中维度 d_j, 用 $\mathrm{CD1}_j$ 和 $\mathrm{CD2}_j$ 分别表示 $\sum x_k^2$ 和 $\sum x_k$, 有

$CD1_j = \sum x_k, CD2_j = \sum x_k^2$. 使用 $CD1_j$ 和 $CD2_j$, 无需历史数据点信息便可计算出质心和判别距离. 这里, 在 1 个簇中只有 1 个或几个点的冷启动的情形下, 仍然需要通过距离阈值来判断新点是否能加入该簇. 当簇中数据点增多, 统计性质稳定时, 再使用判别距离进行判断.

Step 6: 如果 $ds_{\text{index}} > jd_{\text{index}}$, 则转到 Step 7; 否则, 转 Step 8.

Step 7: 建立新簇 CS_{new}, 其中有且仅有 X_t 一个数据点, 转 Step 9.

Step 8: 将 X_t 放入簇 CS_{index}.

Step 9: 移除现有各簇中没有分配到维度的簇, 转 Step 1.

Step 10: 结束.

3　实验分析

3.1　语料数据及实验步骤

由于目前尚无广泛认可的中文文本聚类的公共语料库, 本文采用的实验语料来自搜狐研发中心搜狗实验室的互联网新闻语料 SogouC reduced[8]. 该语料库由不同类别的新闻组成, 能够较好地检验聚类效果. 从该语料库中选取了 5 大类数据 (C000008 财经, C000013 健康, C000014 体育, C000016 旅游以及 C000024 军事). 每类各随机抽取 100, 200, 500, 1 000, 1 500 篇文章, 形成了 S 500, S 1 000, S 2 500, S 5 000, S 7 500 共 5 个语料库, 详细信息如表 1 所示. 分别使用这 5 个语料库的数据对 ID-Squeezer 算法和本文提出的文本数据流聚类算法进行聚类质量和效率比较. 对于 5 个语料库, 均选取 10%的文档作为文本流投影算法初始聚类数据, 实验结果为多次实验取平均值.

表 1　文本数据流聚类测试语料数据

语料库	总类别数	总文档数	词条数	特征词条数	总词数
S 500	5	500	20 340	4 621	119 752
S 1 000	5	1 000	29 732	6 109	240 427
S 2 500	5	2 500	47 130	10 895	611 954
S 5 000	5	5 000	63 845	13 887	1 179 987
S 7 500	5	7 500	76 024	15 681	1 822 811

3.2　聚类结果分析

运行环境设置如下: Thinkpad 笔记本, CPU 双核 2.53 G, 内存 3 G, 磁盘 5 400 rpm, 操作系统 Windows 2003 Server, 测试程序由 C# 语言编写.

目前, 对于聚类结果的评价主要有 3 类方法[9]: 一是基于外部准则的方法, 将聚类结果与已有类别信息结构进行比较; 二是基于内部准则的方法, 将聚类结果与数据集自身特征进行比较; 三是基于相对准则的方法, 对同一算法的不同假设和参数下的聚类结果进行比较. 本实验有每个样本点的真实类别信息, 所以采用常用的 Rand statistic[9], Fowlkes and Mallows Index[9], Normalized Variation of Information[10] 和 Normalized van Dongen Criterion[10]进行评价, 聚类结果的比较如图 1 所示. 由图 1 可以看出, 本文提出的算法在 4 项指标上均优于 ID-Squeezer算法.

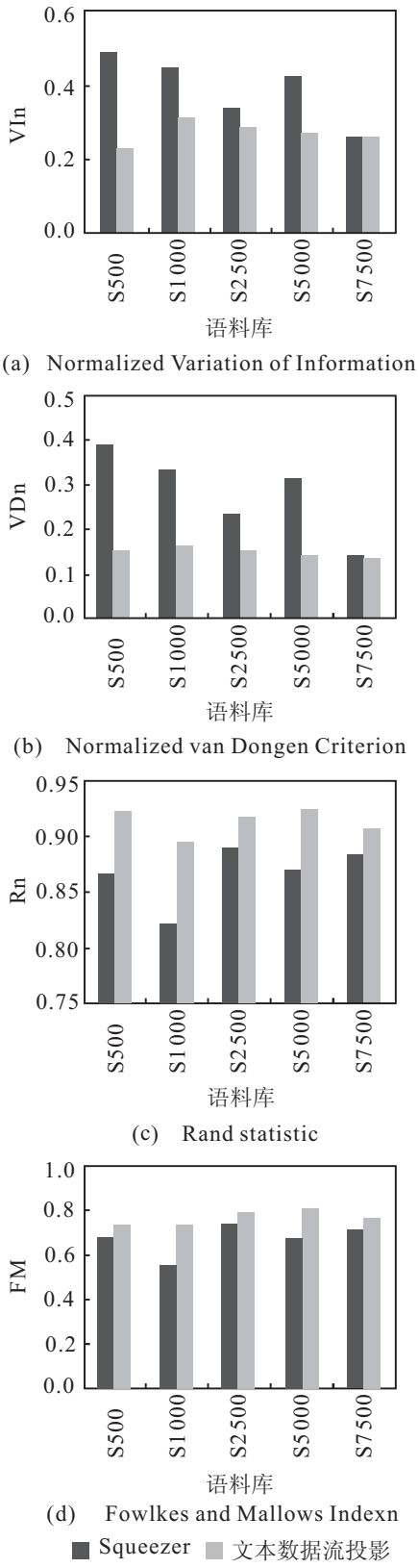

图 1　聚类结果对比分析

在各规模语料库上聚类所需时间对比如图2所示. 由图2可见, 由于存在初始簇的计算和判别距离以及投影维度的计算, 文本数据流投影聚类算法速度稍慢于ID-Squeezer算法, 但文本数据流投影聚类算法的执行时间和数据对象数量仍基本呈线性变化, 具有较好的伸缩性. 总体而言, 文本数据流投影算法以较小的速度代价换来了聚类效果的大幅提升.

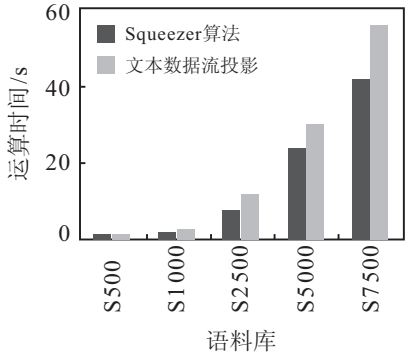

图 2 聚类算法效率比较

使用文本数据流投影聚类算法对数据进行聚类时, 聚类得到的结果集中每簇都有1个维度集, 每个维度是1个词, 每个维度集中词的集合可以作为维度的标签. 例如, S2500采用文本数据流聚类算法的最终维度集如表2所示. 可以看到, 从簇中的词汇可以推测出这个簇所代表的主题.

表 2 聚类维度集结果

类 别	类别维度集
C000008 财经	行情, 个股, 大盘, 利害, 机构, 论坛, 投资者, 股东, 投资, 股价, 板块, 涨停, 风险, 股权, 关系, 资金, 强势, 入市, 有望, 仅供参考, · · ·
C000013 健康	器官, 管理局, 病人, 不适, 药业, 疾病, 企业, 人口, 资料, 答案, 饮料, 外界, 效果, 压力, 比例, 糖尿病, 人体, 局部, · · ·
C000014 体育	比赛, 球队, 队员, 球员, 冠军, 赛季, 联赛, 教练, 决赛, 对手, 胜利, 申花, 球迷, 主场, 俱乐部, 林丹, 比分, 季后赛, 中国队, 李永波, · · ·
C000016 旅游	北京, 特色, 酒店, 单位, 成都, 上海, 森林, 周边, 媒体, 项目, 网络, 欧洲, 杭州, 云南, 泰国, 航空, 风险, 黄金周, 德国, 澳大利亚, · · ·
C000024 军事	导弹, 作战, 军事, 美国, 飞机, 部队, 武器, 空军, 装备, 系统, 发射, 计划, 任务, 研制, 部署, 攻击, 先进, 训练, 军方, 台湾, 战机, · · ·

4 结 论

本文以Squeezer聚类算法为基础, 引入投影聚类算法的思想, 重新定义了类的质心、半径, 引入判别距离的概念, 较好地解决了文本向量的高维、稀疏、多主题问题. 通过开始时间窗数据预处理, 提高了聚类精度; 通过对数据流类的概要信息的提取, 实现了在数据流环境下的高效聚类; 通过对全部向量空间的维度投影, 降低计算复杂度, 同时为聚类结果加上了可理解的语义标签. 通过在网页数据集上的实验, 表明了算法的有效性. 下一步的研究将引入时间衰减因素, 以削弱历史数据的影响, 从而使得聚类结果能够更好地体现时间维度的信息.

参考文献(References)

[1] Aggarwal C C, Yu P S. On clustering massive text and categorical data streams[J]. Knowledge and Information Systems, 2010, 24(2): 171-196.

[2] He Z, Xu X, Deng S. Squeezer: An efficient algorithm for clustering categorical data[J]. J of Computer Science and Technology, 2002, 17(5): 611-624.

[3] 李岩, 王惠文, 叶明, 等. 基于Squeezer算法的大规模矩阵聚类分析[J]. 北京航空航天大学学报, 2009, 35(12): 1499-1502.
(Li Y, Wang H W, Ye M, et al. Modifiable Squeezer cluster algorithm used in large-scale matrix[J]. J of Beijing University of Aeronautics and Astronautics, 2009, 35(12): 1499-1502.)

[4] Aggarwal C C, Han J, Wang J, et al. A framework for projected clustering of high dimensional data streams[C]. Proc of VLDB. Toronto: Morgan Kaufmann, 2004: 852-863.

[5] Aggarwal C C, Wolf J L, Yu P S, et al. Fast algorithms for projected clustering[M]. New York: ACM, 1999: 61-72.

[6] 刘丹. 客户知识管理中的文本挖掘方法与技术研究[D]. 北京: 北京航空航天大学经济管理学院, 2010: 63-64.
(Liu D. Study of text mining in customer knowledge management[D]. Beijing: School of Economics and Management, Beihang University, 2010: 63-64.)

[7] Singhal A. Modern information retrieval: A brief overview[J]. IEEE Data Engineering Bulletin, 2001, 24(4): 35-43.

[8] Internet Web News Corpus from Sogou. SogouC reduced[EB/OL]. [2009-12-01]. http://www.sogou.com/labs/resources.html.

[9] Halkidi M, Batistakis Y, Vazirgiannis M. On clustering validation techniques[J]. J of Interlligent Information Systems, 2001, 17(2/3): 107-145.

[10] Wu J, Xiong H, Chen J. Adapting the right measures for K-means clustering[C]. KDD'09 Proc of the 15th ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining. New York: ACM Press, 2009: 877-885.