文章编号:1000-5862(2014)01-0095-07

基于簇特征的文本增量聚类研究

潘 敏,王明文*,王晓庆,揭安全
(江西师范大学计算机信息工程学院,江西 南昌 330022)

摘要:提出了一种基于簇特征的文本增量聚类算法:充分利用简单、有效的 k-means 算法来进行初始聚类,并保留聚类后每个簇的簇中心、均值、方差、文档数、3 阶中心矩和 4 阶中心矩作为该簇的簇特征,当出现新增数据时,利用初始簇的簇特征对新增数据进行聚类. 在 20newsgroups 数据集上的实验结果表明:相比于对整个数据集进行重新聚类,该算法具有一定的优势.

关键词:增量聚类;文本聚类;中心矩;簇特征
中图分类号:TP 311 **文献标志码**:A

0 引言

当前,随着网络技术与计算机技术的日益发展,互联网已经成为人们获取信息的主要来源之一. 根据中国互联网络发展状况统计报告中的数据显示,互联网上的信息数据量庞大,以指数级的方式增长,并且互联网上的数据大多以文本形式为主. 面对着互联网上信息日益持续爆炸式增长,发现使用传统方法从大规模的数据中获取自身确切需要的信息已越来越难. 因此,如何有效地组织和管理这些数据成为当前急需解决的问题,而文本聚类分析正是一种有效组织和管理文本信息的工具,它能发现大规模数据中潜在的有用模式. 同时,在越来越多的应用中,需要对大规模、高维数据进行处理,有些应用由于数据规模太大,不能一次处理;有些应用中数据库在不断的更新,造成原来的模型对新的数据不适用;有些应用中聚类算法的时间复杂度太高,造成系统开销太大.

针对上述问题,本文提出了一种基于簇特征的文本增量聚类算法,该算法首先充分利用简单、有效的 k-means 算法来进行初始聚类,并保留聚类后每个簇的簇中心、均值、方差、文档数、3 阶中心矩和 4 阶中心矩作为该簇的簇特征,当有新增数据时,利用初始簇的簇特征对新增数据进行聚类. 通过该方法,就无需再对整个数据集重新进行聚类,从而可以降低聚类时间复杂度,提高聚类效率. 在 20newsgroups 数据集上的实验结果也表明:该算法能达到比传统聚类算法更好的聚类效果,且有更高的纯度及更低的时间复杂度;同时,该算法与 Sophoin Khy 等提出的基于语义直方图的增量文本聚类算法的比较结果也说明其具有一定的优势.

1 相关工作

作为一种有效的组织和管理信息的工具,文本聚类被广泛的研究和应用. 目前,普遍使用的文本非增量聚类算法大致可以分为以下几类:基于划分的方法(如 k-means),基于层次的方法(如 HAC),基于密度的方法(如 DBSCAN),基于模型的方法(如 COBWEB),基于网格的方法(如 STING).

为解决在处理大规模、高维数据时传统文本聚类方法表现出的时间复杂度高、效率低等问题,人们相继提出了多种增量聚类算法. Chen Chien-yu 等[1]针对数值数据集提出了一种基于物理中引力理论的层次增量聚类算法,该算法能够达到几乎线性的可扩展性且对输入顺序不敏感. Ian Davidson 等[2]提出了一种高效的增量约束聚类方法,该方法是在聚类的过程中,由用户增量的给出约束条件,相比于使用约束条件进行重聚类的方法,该方法执行效率更高. Sophoin Khy 等[3]提出了一种对最新在线文档进行增量聚类的方法,它使用一个扩展的 k-means 算法进行聚类并对聚类的收敛给出了一个明确的标准. Boris Martínez 等[4]提出了 3 种在模糊模型中基于距

收稿日期:2013-09-17
基金项目:国家自然科学基金(60963014)和江西省自然科学基金(20114BAB201037)资助项目.
通信作者:王明文(1965-),男,江西南康人,教授,博士生导师,主要从事信息检索和并行计算的研究.

离的单步增量聚类算法,该算法不需要事先确定类的数目,且可以检测大小不同的类.

此外,Walaa K. G. 等[5]提出了一种基于语义直方图的增量文本聚类算法,它采取簇间协商的方法来解决文档的插入顺序问题,并且能确保尽可能高的簇凝聚度. Sebastian Luhr 等[6]提出的基于图的增量聚类算法充分利用了代表点之间的连接性,增量地对动态数据流进行聚类. 而 Zhou Yang 等[7]提出的图增量聚类方法,则既考虑了图的拓补结构,又考虑了属性之间的相似度,相比于 2009 年提出的 SA-Cluster[8]图聚类方法,该方法有更低的时间复杂度. Ning Huazhong 等[9]提出的通过引入关联矩阵来更新特征值系统的增量谱聚类算法,它不仅解决了数据点的插入或者删除问题,而且对已有数据点之间的相似度改变进行了处理. Serhat Selcuk Bucak 等[10-12]提出了一种通过非负矩阵分解对视频进行增量聚类的算法,该算法能够标记线性可分和不可分样本.

2 基于簇特征的文本增量聚类模型

本文提出的基于簇特征的文本增量聚类模型是由 2 个阶段组成的,分别为初始聚类阶段和增量聚类阶段. 图 1 给出了 2 阶段模型的增量聚类过程.

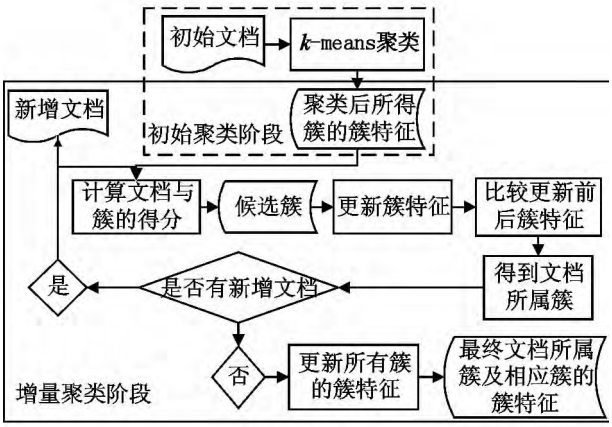

图 1 基于簇特征的文本增量聚类流程图

2.1 初始聚类阶段

首先是对初始部分文档进行预处理; 然后使用快速、高效的 k-means 聚类算法进行聚类,聚类过程中,本文采用余弦值计算文档与簇中心之间的相似度,即

$$sim(d_n, CCenter_k) = \cos(d_n, CCenter_k), \quad (1)$$

其中 $\cos(d_n, CCenter_k)$ 为初始文档 d_n 与簇 C_k 的簇中心(用 $CCenter_k$ 表示)之间的余弦相似度,$1 \leqslant n \leqslant N$,$1 \leqslant k \leqslant K$; 之后,对聚类后得到的 k 个簇,均保留其簇中心、均值、方差、文档数、3 阶中心矩和 4 阶中心矩作为该簇的簇特征来代表该簇,同时保留聚类结果,包括纯度、熵和归一化互信息. 为了更好地理解簇特征,本文将每个簇 C_k 的簇特征 CF_k,用下列集合的形式进行表示: $CF_k = \{CCenter_k, DocNum_k, Mean_k, Var_k, ThirdCM_k, FourCM_k\}$, 集合中的元素依次代表的是簇中心、文档数、均值、方差、3 阶中心矩和 4 阶中心矩,除文档数 $DocNum_k$ 外,其余簇特征的计算方法为

$$CCenter_k = \sum_{d_j \in C_k} d_j / DocNum_k, \quad (2)$$

$$Mean_k = \sum_{j=1}^{DocNum_k} dis(d_j, CCenter_k) / DocNum_k, \quad (3)$$

$$Var_k = \sum_{j=1}^{DocNum_k} (dis(d_j, CCenter_k) - Mean_k)^2 / DocNum_k, \quad (4)$$

$$ThirdCM_k = \sum_{j=1}^{DocNum_k} (dis(d_j, CCenter_k) - Mean_k)^3 / DocNum_k, \quad (5)$$

$$FourCM_k = \sum_{j=1}^{DocNum_k} (dis(d_j, CCenter_k) - Mean_k)^4 / DocNum_k, \quad (6)$$

其中 $DocNum_k$ 为属于簇 C_k 的文档数; $dis(d_j, CCenter_k) = \sqrt{\sum_{x \in d_j, y \in C_k} (x - y)^2}$ 为文档 d_j 与簇 C_k 的簇中心之间的欧几里得距离,x,y 分别表示属于文档 d_j 和簇 C_k 的词项.

2.2 增量聚类阶段

首先是对增量部分文档进行预处理,方法与初始聚类阶段一样; 然后,利用初始阶段保留的簇特征,对每一个新增文档 d_m,计算该文档与现有簇的得分

$$score(d_m, CCenter_k) = (1 - \lambda) sim(d_m, CCenter_k) - \lambda dis(d_m, CCenter_k), \quad (7)$$

其中 λ 为权重因子,取值在 0 到 1 之间,$1 \leqslant m \leqslant M$; 之后,选取得分最高的簇为

$$C_{\max} = \arg \max_{1 \leqslant k \leqslant K} score(d_m, CCenter_k), \quad (8)$$

并记录下该簇的簇特征,记为 CF_{old},之后将此文档放入该簇中,同时更新该簇的簇特征,记为 CF_{new},比较 CF_{old} 与 CF_{new},如果满足下列 3 个条件中的任意一个: (ⅰ) $Mean_{\text{new}} < Mean_{\text{old}}$,(ⅱ) $Var_{\text{new}} < Var_{\text{old}}$ 且 $Mean_{\text{new}} < Mean_{\text{old}}$,(ⅲ) $ThirdCM_{\text{new}} < ThirdCM_{\text{old}}$ 且 $FourthCM_{\text{new}} < FourthCM_{\text{old}}$,则将此文档放入该簇中,并设置其簇标为该簇的簇标; 否则,将此文档的

簇标设置为 $K+1$; 循环处理,直至所有新增文档均处理完毕; 之后,检查是否有文档的簇标为 $K+1$,若有,则创建一个空簇 C_{K+1},将所有簇标为 $K+1$ 的文档放入 C_{K+1} 中,同时簇个数加 1: $K=K+1$; 最后,更新这 K 个簇的簇特征,同时,保留新增文档的聚类结果. 以后每次增量聚类过程都与此一致.

2.3　算法与分析

为了更好地将本文提出的算法与传统聚类算法的时间复杂度进行对比分析,图 2 给出了本文提出算法的具体执行步骤.

算法中各个符号所代表的含义参照本文 2.1 及 2.2 节中的定义. 由于本文的增量聚类分 2 个阶段进行,因而需要分开来计算该算法的时间复杂度.

先对初始聚类阶段,时间复杂度为 $O(nkt)$,其中 k 为初始聚类簇数目,n 为初始文档数目,t 为算法的迭代次数; 再对增量聚类阶段,时间复杂度为 $\sum_{i=1}^{m} O(k_i \Delta n)$,其中 k_i 为执行第 i 次增量聚类时已有的簇数目; Δn 为每次需要执行增量聚类的文档数,即每次新增文档数(由于 20 个类别中,最后一次新增文档数为 1 997,其余每次都是 2 000,因而为了计算的方便,在这里假设每次新增文档数是一样的); m 为增量聚类总的执行次数; 因此,本文提出的增量聚类算法时间复杂度为 $O(nkt)+\sum_{i=1}^{m} O(k_i \Delta n)$. 由于每次增量聚类后簇的数目可能会有所增加,也即 $k_i \geqslant k$,但这并不会造成算法时间复杂度的增加,因而可以得出: $\sum_{i=1}^{m} O(k_i \Delta n) \approx \sum_{i=1}^{m} O(k \Delta n) = O(mk \Delta n)$,故最终算法的时间复杂度大约为 $O(nkt)+O(mk \Delta n)$. 而使用 k-means 聚类算法的时间复杂度为 $O(nkt)+O((n+\Delta n)kt)+O((n+2\Delta n)kt)+\cdots+O((n+m\Delta n)kt)=O((m+1)nkt)+O((m+1)mkt\Delta n/2)$,即 $O((m+1)(n+m\Delta n)kt)$,相比之下,本文提出算法的时间复杂度更小.

初始聚类阶段算法

输入: N 个初始聚类文档,聚类簇数 K;
输出: N 个文档的聚类结果及每个簇的簇特征;
(ⅰ) 随机选取 K 个文档作为 $CCenter_k$;
(ⅱ) 迭代 5 次;
　重复
　　使用(1)式计算 $sim(d_n, CCenter_k)$,并更新 d_n 簇标为相似度最高的簇的簇标;
　直到 N 个文档处理完毕;
　更新;
(ⅲ) 使用公式(2) ~ (6) 计算;
(ⅳ) 存储,并保留聚类结果,包括纯度、熵及归一化互信息.

增量聚类阶段算法

输入: M 个新增文档,K 个簇的簇特征;
输出: M 个文档的聚类结果及聚类后得簇的簇特征;
(ⅰ) 重复
　(a) 使用公式(7) 计算 $score(d_m, CCenter_k)$;
　(b) 使用公式(8) 选取 $C_{\max}$,并获得 CF_{old};
　(c) 更新簇特征为 CF_{new};
　(d) 比较 CF_{old} 与 CF_{new},同时更新 d_m 簇标;
(ⅱ) 直到 M 个文档处理完毕;
(ⅲ) 判断 d_m 中是否有簇标为 $K+1$,若有,建一空簇 C_{K+1},且簇个数加 1: $K=K+1$;
(ⅳ) 更新并存储 CF_K,保留 M 个文档的聚类结果.

图 2　本文提出的算法

3　实验设计及结果分析

3.1　实验数据准备

本文使用 20newsgroups(20 个新闻组,英文数据集) 进行实验. 该数据集总共有 20 个类,除类别 soc. religion. christian 为 997 篇外,其余每个类有 1 000 篇,共 19 997 篇文档. 在进行实验前,首先需要对文档数据进行预处理,为了确保实验结果的可比性,本文对增量及非增量聚类的文本数据,采用相同的预处理方法.

3.2　实验设计

本文主要进行了 2 次实验: (ⅰ) 对随机选取的 10 个类别进行的实验; (ⅱ) 对全部 20 个类别的实验. 为了验证本文提出方法的有效性及可行性,每次实验都进行 2 组实验: 一组为文本增量聚类实验,另一组为文本非增量聚类实验. 每次实验的数据构造如下: (i) 10 个类别: (a) 对文本增量聚类: 初始聚类阶段,聚类文本数为 2 000 篇(每个类别 200 篇,采取随机选取的方式); 增量聚类阶段,每次增量数为 500 篇文档(每个类别 50 篇),共进行 16 次; (b) 对文本非增量聚类: 利用 k-means 分别对 2 500, 3 000, …, 10 000 篇文档聚类,共 16 次; (ii) 20 个类别: (a) 对文本增量聚类: 初始聚类阶段,聚类文本数为 2 000 篇(每个类别 100 篇,随机选取方式); 增量聚类阶段,除最后一次增量数为 1 997 篇之外,其余每次为 2 000 篇(每个类别 100 篇),共需进行 9

次;(b) 对文本非增量聚类:利用 k-means 分别对 4 000,6 000,…,9 997 篇文档进行聚类,共 9 次;

3.3 评价指标

本文使用纯度(*Purity*)、熵(*Entropy*)和归一化互信息(Normalized Mutual Information,NMI)作为评价的指标,来衡量提出方法与传统方法的优劣.

纯度的定义为

$$Purity = \sum_{i=1}^{n_i} \max_j(N_i^j) / N, \quad (9)$$

其中 n_i 为预定义类别的个数,N_i^j 表示聚类 j 中包含类别 i 中的文档的个数,N 为总文档数. 由于在本文中每进行一次增量聚类,都会得到一个聚类纯度,因而纯度的计算方法与传统的方法有所区别,其为 $Purity_n = (N_0 \times P_0 + \sum_{i=1}^{n} N_i \times P_i) / (N_0 + \sum_{i=1}^{n} N_i)$,其中 P_0,N_0 分别为初始聚类阶段的纯度和文档个数,P_0 的计算方法同(9)式,$N_0 = 2\ 000$;P_i 为第 i 次增量聚类后 N_i 个文档的纯度;$Purity_n$ 为第 n 次增量聚类后全部 $N_0 + \sum_{i=1}^{n} N_i$ 个文档的纯度;对 10 个类别,$1 \leqslant n \leqslant 16$,$N_i = 500$;对 20 个类别,$1 \leqslant n \leqslant 9$,除 $N_9 = 1\ 997$ 之外,其余 $N_i = 2\ 000$.

熵的定义为

$$Entropy = -\sum_{c_i \in C} \sum_{c_j \in C'} p(c_i, c_j) \log p(c_i, c_j) N_i / N, \quad (10)$$

其中 N_i 为类别 i 中的文档数,N 为总文档数,$p(c_i, c_j)$ 表示簇 c_i 与簇 c_j 的共现概率. 与纯度一样,使用增量聚类算法进行聚类时,熵的计算方法为 $Entropy_n = (Entropy_{n-1} \times (N_0 + \sum_{i=1}^{n-1} N_i) + E_n \times N_n) / (N_0 + \sum_{i=1}^{n} N_i)$,其中 $Entropy_0$,N_0 分别表示初始聚类阶段的熵和文档个数,$Entropy_0$ 的计算方法同(10)式;$Entropy_n$,E_n 分别表示第 n 次增量聚类后全部 $N_0 + \sum_{i=1}^{n} N_i$ 和 N_n 个文档的熵;n 和 N_i 的取值与纯度中一致. 归一化互信息的计算方法类似于熵的计算方法,这里就不赘述了.

3.4 实验结果及分析

本文使用快速、高效的 k-means 聚类算法作为基准方法与本文提出的算法(记为 TICBCF,Text Incremental Clustering Based on Cluster Features)进行比较,同时,也将TICBCF方法与基于语义直方图的文本增量聚类算法(SHC)进行了比较. 最终的比较结果表明本文提出的 TICBCF 方法具有一定的优势.

为消除初始点对 k-means 聚类算法的影响且鉴于该算法的不稳定性,大多数文献,如文献[3],均采取多次运行 k-means 算法所得结果的平均值作为该算法的聚类结果,并将其与文中提出算法的结果进行比较. 与文献[3]类似,本文也选择多次运行 k-means 算法所得结果的平均值作为该算法的聚类结果(记为 k 平均),并将其与 TICBCF 进行比较. 为进一步表明本文提出算法的好处,本文将多次运行 k-means 算法所得结果中的最优结果作为该算法的另一种聚类结果(记为 k 最优),并将其与 TICBCF 进行比较,这在其它文献中,是并没有给出的. 在本文中,对同一部分文档,只运行 5 次 k-means 算法.

表 1 给出了使用上述 3 种方法对 10 个类别文档进行聚类后得到的纯度、熵及归一化互信息的对比结果.

表 1 3 种方法对 10 个类别结果的影响

文档数	k 最优 *Purity*	TICBCF *Purity*	k 平均 *Purity*	k 最优 *Entropy*	TICBCF *Entropy*	k 平均 *Entropy*	k 最优 NMI	TICBCF NMI	k 平均 NMI
2 000	0.577	0.577	0.558	1.849	1.849	1.890	0.443	0.443	0.417
2 500	0.558	0.595	0.521	1.766	1.764	1.868	0.468	0.469	0.437
3 000	0.549	0.615	0.521	1.654	1.669	1.851	0.502	0.497	0.442
3 500	0.664	0.629	0.568	1.463	1.596	1.737	0.559	0.519	0.476
4 000	0.584	0.641	0.501	1.696	1.547	1.867	0.489	0.534	0.437
4 500	0.562	0.648	0.533	1.773	1.506	1.777	0.466	0.546	0.464
5 000	0.547	0.654	0.485	1.802	1.483	1.839	0.457	0.553	0.446
5 500	0.539	0.658	0.512	1.760	1.460	1.840	0.470	0.560	0.445
6 000	0.609	0.663	0.524	1.622	1.442	1.730	0.511	0.565	0.479
6 500	0.525	0.667	0.478	1.694	1.423	1.823	0.489	0.571	0.450
7 000	0.624	0.674	0.588	1.502	1.398	1.530	0.547	0.578	0.539

续表 1

文档数	k 最优 *Purity*	TICBCF *Purity*	k 平均 *Purity*	k 最优 *Entropy*	TICBCF *Entropy*	k 平均 *Entropy*	k 最优 NMI	TICBCF NMI	k 平均 NMI
7 500	0.549	0.675	0.486	1.674	1.389	1.800	0.495	0.581	0.458
8 000	0.546	0.677	0.476	1.565	1.377	1.844	0.528	0.585	0.444
8 500	0.615	0.679	0.558	1.561	1.366	1.626	0.530	0.588	0.510
9 000	0.588	0.681	0.543	1.610	1.354	1.667	0.515	0.592	0.497
9 500	0.512	0.684	0.487	1.819	1.345	1.860	0.452	0.595	0.439
10 000	0.507	0.686	0.495	1.730	1.334	1.732	0.479	0.598	0.478

从表 1 可以看出,使用 TICBCF 方法进行增量聚类得到的结果要明显高于使用 k-means 平均方法得到的结果; 而且,除了文档总数为 3 500 之外,TICBCF 方法得到的结果也是高于 k-means 最优方法得到的结果. 出现上述情况,主要是因为 TICBCF 方法只需对新增文档进行增量聚类,且每次新增文档数相对于簇中已有文档数来说,所占比例不大,故对整体而言,造成的影响较小,同时,在选择新增文档所属簇时,使用本文提出的结合了相似度及距离值的方法来计算簇与文档之间的得分,并对得分最高的簇,比较加入前与加入后簇特征的改变量来最终确定该文档所属的簇,这比仅仅使用相似度或者距离来确定文档所属簇的方法有更高的准确度; 而 k-means 方法是要对所有的文档重新聚类,同时该方法又极易受到初始点选择的影响,因而在某些情况下,使用 k-means 方法聚类会得到更好的聚类结果,但也会得到更差的聚类结果.

表 2 给出了使用上述 3 种方法对 20 个类别文档进行聚类后得到的纯度、熵及归一化互信息的对比结果.

表 2　3 种方法对 20 个类别结果的影响

文档数	k 最优 *Purity*	TICBCF *Purity*	k 平均 *Purity*	k 最优 *Entropy*	TICBCF *Entropy*	k 平均 *Entropy*	k 最优 NMI	TICBCF NMI	k 平均 NMI
2 000	0.560	0.560	0.500	2.033	2.033	2.190	0.529	0.529	0.493
4 000	0.579	0.591	0.532	1.699	1.850	2.009	0.606	0.571	0.534
6 000	0.599	0.621	0.567	1.712	1.688	1.785	0.603	0.609	0.586
8 000	0.649	0.638	0.563	1.360	1.591	1.779	0.685	0.631	0.588
10 000	0.654	0.651	0.541	1.405	1.525	1.866	0.674	0.646	0.568
12 000	0.591	0.660	0.537	1.639	1.468	1.890	0.620	0.660	0.562
14 000	0.571	0.666	0.548	1.767	1.439	1.873	0.591	0.667	0.566
16 000	0.579	0.671	0.542	1.748	1.409	1.864	0.595	0.673	0.568
18 000	0.686	0.676	0.590	1.276	1.377	1.641	0.704	0.681	0.620
19 997	0.616	0.680	0.545	1.711	1.350	1.920	0.604	0.687	0.555

从表 2 可以看出,对全部 20 个类别进行聚类时,使用 TICBCF 方法得到的结果仍然要高于 k-means 平均方法的结果; 另外,也可以看出,在文档总数分别为 8 000、10 000 和 18 000 时,使用 TICBCF 方法得到的结果略逊于 k-means 最优方法的结果; 而在文档总数为 4 000 的时候,虽然 TICBCF 方法得到的纯度结果要高于 k-means 最优方法的结果,但是熵和归一化互信息的结果却更差了. 这是因为全部 20 个类别时,每次新增文档数从 500 增长为 2 000,文档携带的信息也随之增加,因而相对于簇中已有文档数来说,所占比例增加,故对整体而言,造成的影响较大,同时出现异常文档的机会也相应地增加,而且簇与簇之间的区别也越来越不明显; 而 k-means 受初始点选择的影响较大,因而偶尔出现聚类结果好的情况是十分有可能的,但就总体而言,TICBCF 方法得到的结果还是要高于 k-means 方法的结果.

图 3 给出了权重因子 λ 对 10 个类别的实验结果的影响. 它表明,当 λ 的取值为 0 至 0.02 之间时,均能得到比较好的聚类结果. 但随着 λ 取值地逐渐增大,实验结果也开始逐步地下降,这是因为文档与簇之间的距离值增大,而相似度值减小,使得文档与簇之间的得分减小,进而造成文档的错分. 对 10 个类别,本文中取 $\lambda = 0.01$ 的聚类结果与 k-means 的聚类结果进行比较. 对 20 个类别,λ 的取值亦为 0.01. 由于篇幅关系,之后的实验也只给出 10 个类别纯度的结果,熵及归一化互信息的结果与纯度的结果类似.

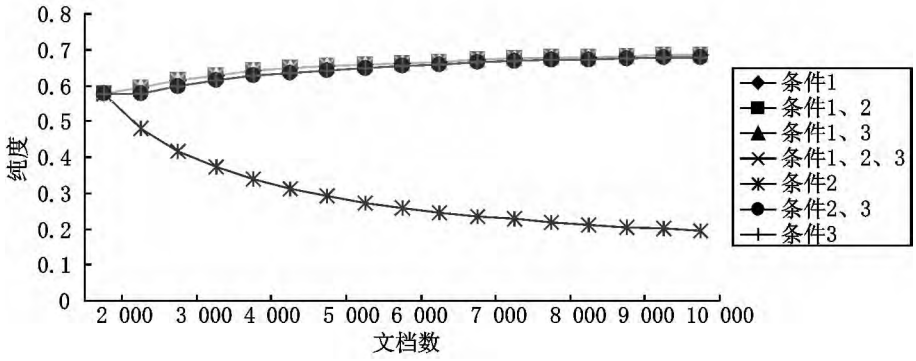

图 3 λ 对 10 个类别纯度的影响

图 4 给出了实验中 3 个条件对 10 个类别聚类结果的影响. 从图 4 可以看出: 单独使用条件 1 或是与另外 2 个条件的任意组合所得的结果是一样且最好的; 单独使用条件 3 或是结合条件 2 所得的结果略差些; 而单独使用条件 2 所得的结果最差. 由此可见,本文中所用的 20 newsgroups 数据集中的文档分布比较均匀,因而使用方差进行增量聚类的结果最差,而使用 3 阶中心矩及 4 阶中心矩能达到与使用均值进行增量聚类相当的结果. 因此,在本文中,使用方差进行聚类的结果并不是很好,但使用均值、3 阶中心矩和 4 阶中心矩进行聚类的结果比较理想,因而,在今后的实验中,可以考虑采取其它的方式,比如利用方差与均值的比值作为一个条件来对文档进行聚类.

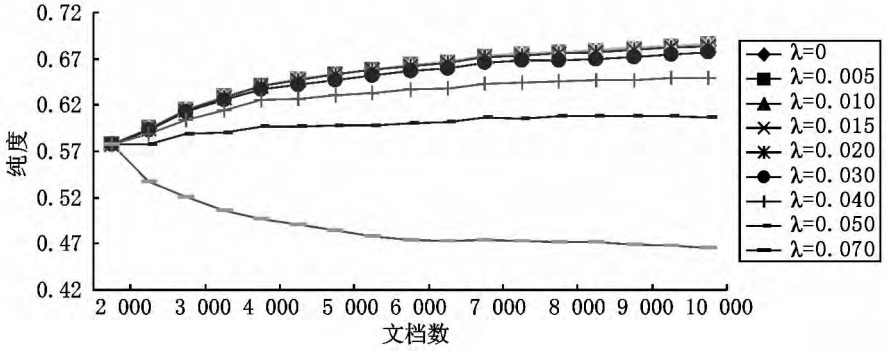

图 4 3 个条件对 10 个类别纯度的影响

文本增量聚类算法也已有比较多的研究,本文选择其中较新的一种方法(文献[3]提出的 SHC 方法)与 TICBCF 方法进行比较. 使用 SHC 对聚类纯度的提高比率大约为 25.76%,对熵的最多降低比率约为 38.18%,平均约为 24.24%; 而使用 TICBCF 对纯度的提高比率约为 21.20%,略低于 SHC,但对熵的最多降低比率却要高于 SHC,约为 42.17%,在大多数情况下,对熵的降低比率达到 28.00% 以上,且从熵的平均降低比率(约 22.33%)来看,本文方法与文献[3]中方法差异并不大. 出现这些情况是由于(i)文献[3]中使用的数据集是从 20newsgroups 中抽取组合而成的,并对这些数据进行 4 组实验,每组实验包含 4 个类别的数据,且文档总数分别为 400,400,300 和 300 篇,共 1 400 篇文档,其中: 第 1 组和第 3 组实验数据为不相关文档,第 2 组和第 4 组为相关文档,第 1 组和第 2 组每个类别的文档数相同,第 3 和第 4 组不相同,而本文进行的 2 组实验分别包含 10 个和 20 个类别,且文档总数分别为 10 000和 19 997 篇,其中 10 个和 20 个类别的数据都是随机抽取的,10 个类别中每个类别的文档数相同,20 个类别中,尽管最后一个类别文档数有所不同,但与其它类别基本保持平衡; (ii)文献[3]中的文档总数最多为 400 篇,因而每次执行增量聚类的文档数大大少于本文的文档数(10 个类别为 500 篇,20 个类别为 2 000 或 1 997 篇),这也使得文档的错分率更小,进而纯度要高于本文方法,且由于文献[3]中执行增量聚类的次数不多于 4 次,而本文执行增量聚类的次数分别为 9 次和 16 次,因而本文中对纯度的总体提高比率要略低些,但对熵的总体降低比率却更好.

总之,本文提出的 TICBCF 方法不仅能够达到比传统聚类算法更好的聚类效果,而且有更高的纯度及更低的时间复杂度,且与其它的增量聚类算法的比较结果也表明该方法具有一定的优势.

4　结论

增量聚类算法,是能够在已有聚类结果的基础之上,通过对新增数据逐个或者批量进行处理,避免大量重复计算,同时,在数据不断增长的情况下,利用增量聚类算法,不仅易于维护和扩充聚类结果,而且能够提高聚类效率. 但是,如何保证增量聚类算法能达到传统聚类算法的效果是一个十分值得研究的问题,对此,本文提出了一种基于簇特征的文本增量聚类算法,该算法首先利用 k-means 进行初始聚类,只保留聚类后得到的每个簇的簇特征,原有的文档不再保留,从而可以节省存储空间; 当出现新的文档时,只需要利用这些簇的簇特征与它们进行聚类,而不需要将它们与之前原有的文档进行重新聚类. 实验结果也表明,该算法不仅能够提高聚类的准确度及效率,降低时间复杂度,而且能够取得比传统文本聚类算法更好的聚类效果,且与已有的一些方法相比,本文提出的算法也具有一定的优势.

文本增量聚类非常具有实用和研究价值,未来的主要工作有以下几个方面: (i) 选择其它更好的聚类方法作为初始的聚类方法; (ii) 利用更大的文本数据集进行实验,进一步验证该方法的有效性; (iii) 考虑能否保留其他的簇特征,以便更好的与新增文档进行增量聚类,进而提高聚类准确度.

5　参考文献

[1] Chen Chien-yu, Hwang Shien-ching, Oyang Yen-jen. A statistics-based approach to control the quality of subclusters in incremental gravitational clustering [J]. Pattern Recognition, 2005, 38(12): 2256-2269.

[2] Davidson Ian, Ravi S S, Ester Martin. Efficient increment-al constrained clustering [EB/OL]. [2013-03-12]. http: // www. cs. ucdavis. edu/ ~ davidson/Publications/KDDinc. pdf.

[3] Khy Sophoin, Ishikawa Yoshiharu, Kitagawa Hiroyuki. Incremental clustering based on novelty of on-line documents [J]. Nihon Detabesu Gakkai Letters, 2006, 5(1): 57-60.

[4] Martínez Boris, Herrera Francisco, Fernández Jesús, et al. An incremental clustering method and its application in online fuzzy modeling [J]. Studies in Fuzziness and Soft Computing, 224: 163-178.

[5] Walaa K G, Mohamed S K. Incremental clustering algorithm based on phrase-semantic similarity histogram [EB/OL]. [2013-03-17]. http: // ieeexplore. ieee. org/xpl/abstractKeywords. jsp? arnumber = 5580499.

[6] Luhr Sebastian, Lazarescu Mihai. Incremental clustering of dynamic data streams using connectivity based representative points [J]. Data & Knowledge Engineering, 2009, 68(1): 1-27.

[7] Zhou Yang, Cheng Hong, Jeffrey X Y. Clustering large attributed graphs: an efficient incremental approach [EB/OL]. [2013-03-19]. http: // ieeexplore. ieee. org/xpls/abs_all. jsp? arnumber = 5694023.

[8] Zhou Yang, Cheng Hong, Jeffrey X Y. Graph clustering based on structural/attribute similarities. VLDB, pp. 718-729, 2009.

[9] Ning Huazhong, Xu Wei, Chi Yun, et al. Incremental spectral clustering by efficiently updating the eigen-system [J]. Pattern Recognition, 2010, 43(1): 113-127.

[10] Bucak Serhat Selcuk, Gunsel Bilge. Incremental clustering via nonnegative matrix factorization [EB/OL]. [2013-03-19]. http: // ieeexplore. ieee. org/xpls/abs_all. jsp? arnumber = 4761104.

[11] 曾道建,来斯惟,张元哲,等. 面向非结构化文本的开放式实体属性抽取 [J]. 江西师范大学学报: 自然科学版, 2013, 37(3): 279-283, 305.

[12] 万中英,王明文,揭安全,等. 投影寻踪模型中投影指标的改进 [J]. 江西师范大学学报: 自然科学版, 2013, 37(3): 284-287.

A Research on the Text Incremental Clustering Based on Cluster Features

PAN Min, WANG Ming-wen*, WANG Xiao-qing, JIE An-quan

(College of Computer Information Engineering, Jiangxi Normal University, Nanchang Jiangxi 330022, China)

Abstract: A text incremental clustering algorithm based on cluster features has been presented. Firstly, initial clustering is performed by making full use of simple and efficient k-means algorithm. Secondly, the clustering center, mean, variance, the number of document, the third central moment and the fourth central moment are saved as the cluster features of each cluster. Finally, when new documents occur, they are incrementally clustered with those cluster features. The experimental results on 20newsgroups data set demonstrate that the algorithm the paper presents has some advantages.

Key words: incremental clustering; text clustering; central moment; cluster features

(责任编辑: 冉小晓)