一种用于文本聚类的改进二分 K-均值算法

邹 海,李 梅

(安徽大学 计算机科学与技术学院,安徽 合肥 230039)

摘 要:在已有聚类算法的基础上,提出了一种新的文本聚类新方法——合作二分 K-均值算法(简称 CBKM)。该算法以 K-均值算法和二分 K-均值算法为基础,通过整体聚类、合作聚类和聚类融合 3 个阶段,对中间聚类结果进行再次划分,产生了具有更好聚类效果的集合。实验结果表明,合作二分 K-均值算法的聚类性能优于 K-均值算法和二分 K-均值算法。

关键词:向量空间模型;融合因子;合作聚类

中图分类号:TP274 **文献标识码**:A **文章编号**:1674-7720(2010)12-0064-04

Optimize bisecting K-means algorithem applied in text clustering

ZOU Hai, LI Mei

(Institute of Computer Science & Technology, Anhui University, Hefei 230039, China)

Abstract: Based on the local clustering algorithems, we propose a new clustering algorithem cooperative bisecting K-means algorithem(short for CBKM) which combines both KM algorithem and BKM algorithem together. The CBKM re-cluster the intermediate result for a better clusters in three phases, the global phase, the cooperative clustering phase and the merging phase. Experimental shows that the performance of CBKM is better than both KM algorithem and BKM algorithem.

Key words: VSM; merging cohesiveness factor; cooperative clustering

聚类[1]是将物理或抽象对象集合按有关特性的相似程度进行分组的过程。聚类产生每一组数据称为一个类,类中每一个数据称为一个对象。聚类的目的是使同一簇中对象的特性尽可能地相似,而不同簇对象之间的差异尽可能地增大。聚类作为一种无监督的学习方法,能从数据集中发现数据的分布情况,是一种强有力的信息处理方法。随着英特网的普及,各种文本资源呈爆炸式的增长,用户迫切需要高效率的信息检索,在这种情况下,文本聚类也得到越来越多的重视。

K-均值算法是最常用的文本聚类方法,其优点是时间复杂度小,易于实现。但 K-均值得到的只是局部最优解,而不是全局最优解,聚类效果较差。二分 K-均值[2]算法能够在一定程度上改进 K-均值算法,得到较好的聚类效果,但是在某些情况下,二分 K-均值算法容易产生成员碎片,由于这些碎片不能通过其他方法进行再次聚类而分到某个类中,从而影响系统的总体性能。本文对二分 K-均值算法进行改进,提出了一种新的聚类算法合作二分 K-均值算法。实验结果表明:合作二分 K-均值比 K-均值算法和二分 K-均值算法具有更好的聚类效果。

1 向量空间模型(VSM)和相似度的计算

向量空间模型[3]是目前用的较多的一种文档表示方法,其基本思想是把用于挖掘的页面文档表示成为一个文档集合 $D=\{D_1, D_2, \cdots, D_n\}$。其中每一篇文档 D_i 看做是一组词条 $(t_1, t_2, \cdots, t_n)$ 组成的集合,对于词条 t_i 根据其在文档中的重要程度赋予一定的权重 w_i,这样文档 D_j 就表示为:

$$D=\{(t_1, w_1), (t_2, w_2), \cdots, (t_n, w_n)\} \tag{1}$$

根据 TF-IDF 算法,每个特征向量 t_i 在文档 D_j 中的权重 $W_i=tf_i \times idf_i$,tf_i 为该词条在文档中出现的频率,idf_i 为与文档集合中该词条的文档的数目有关的函数,一般定义 idf_i 为:

$$idf_i=\log_2 \frac{N}{DF(t_i)}+0.01 \tag{2}$$

其中,N 为文档总数,$DF(t_i)$ 为包含词条 t_i 的文档数。一般,W_i 在 $[0, 1]$ 之间,故将其规范化为:

$$W_i^*=\frac{tf_i \times idf_i}{\sqrt{\sum_{k=1}^{N}(tf_{ik})\times \log_2^2 \frac{N}{DF(t_i)}+0.01}} \tag{3}$$

文本采用了向量空间模型表示以后，文本之间的相似度就能够进行量化。本文中相似度的计算是采用公认的相似度计算效果比较好的夹角余弦函数：

$$\mathrm{Sim}(X,Y)=\frac{\sum_{i=1}^{m} W_{Xi}\times W_{Yi}}{\sqrt{\sum_{k=1}^{m}(W_{xi})^2\times\sum_{k=1}^{m}(W_{yi})^2}} \tag{4}$$

2 K-均值(KM)算法

K-均值算法是文本聚类中最常用的一种方法，其算法描述过程为：

(1)对于给定的聚类数目 K，首先随机选择 K 个文本作初始的类质心 $C_1,C_2,\cdots,C_k$；

(2)计算每个文本与各个类质心的相似度，将它赋给最相似的类，然后重新计算每个类的质心：$centorC_i=\frac{\sum_{di\in ci} d_k}{n}$；

(3)不断迭代以上过程步骤(2)，直到类中心不再改变。

3 二分 K-means(BKM)算法

BKM 算法是 KM 算法的一个变形，其具体算法为：

(1)输入聚类数目 K；

(2)随机选取 2 个文本 C_1、C_2 作为初始类质心，用 K-均值算法对文本集进行二分聚类，得到包含 2 个类的聚类 $v1$、$v2$；

(3)根据一定的算法，在这两个数据集中选出包含类中成员个数最大的一个类(如 $v1$)再次进行二分聚类，得到聚类 $v3$、$v4$；

(4)反复迭代以上步骤(2)、(3)，直到聚类中类的个数等于 K。

4 合作二分 K-均值算法(CBKM 算法)

事实表明，BKM 算法比 KM 算法的聚类效果好，它的性能几乎可以达到层次聚类算法的性能。但是在某些情况下，BKM 算法容易产生成员碎片，这些碎片不能通过其他方法进行再次聚类而分到某个类中，从而影响系统的总体性能。为此本文提出了一种新的算法——合作二分 K 均值(简称 CBKM)算法。CBKM 算法把 BKM 算法中通过分类两个类中的一个类而生产层次二叉树的方法和 KM 中的同步中间合作方法结合起来。CBKM 算法首先将整个数据集看成一个类，再对其进行划分生成一棵二叉树。其生成二叉树的过程分为以下 3 个阶段：整体聚类阶段，合作聚类阶段，融合阶段。

4.1 整体聚类阶段

在整体聚类阶段，分别用算法 1 和算法 2 对同一文本集合 l 进行聚类，得到具有 l 个类的两个集合 $S^{BKM}(l)$、$S^{KM}(l)$，可分别表示为：

$$S^{BKM}(l)=\{S_j^{BKM},0\leqslant j\leqslant l-1\},l=2,3,\cdots,k \tag{5}$$

$$S^{KM}(l)=\{S_j^{KM},0\leqslant j\leqslant l-1\},l=2,3,\cdots,k \tag{6}$$

这两个集合是下一阶段合作聚类的基础。

4.2 合作聚类阶段

合作聚类阶段的主要任务是构建一个合作应变矩阵(简称 *CCM*)。*CCM* 是一个二维的大小为 $L\times L$ 的矩阵，通过整体聚类阶段产生的两个聚类集合构建。其中的任一元素定义为：

$$\boldsymbol{CCM}(S_i,S_j)=\{S_i\in S^{KM}(l)\wedge S_j\in S^{BKM}(l)\wedge S_i\in S_j\}。$$

经过此步骤，文本集 $S^{BKM}(l)$ 和 $S^{KM}(l)$ 中的每一个聚类就被划分成更小的不相交的集合。称这些集合为子类：$Sb=\{Sb_i\}_{i=0}^{n_{sb}-1}$，其中 n_{sb} 为子类的个数。可以推知，n_{sb} 的上限为 *CCM* 矩阵中元素的个数 L^2。

4.3 融合阶段

在融合阶段，利用在合作聚类阶段得到的合作应变矩阵，重新得到包含 L 个类的聚类。这个聚类有着比采用 KM 和 BKM 算法得到的聚类更好的性能。首先构造一个 $n_{sb}\times n_{sb}$ 的对称矩阵 *CCM*。由于 *CCM* 的对称性，只需要存储其上三角或者是下三角。矩阵中第 i 行第 j 列个元素代表子类 Sb_i 和子类 Sb_j 的融合因子：$msf(Sbi,Sbj)$。这个融合因子是根据每个子类中元素的成对相似性决定的。每个子类 Sb_i 中元素的成对相似性用一个相似柱状图表示。

4.3.1 相似柱状图

相似柱状图 H_i 是表示子类 Sb_i 中元素之间相似性的一个分布图。相似柱状图[4]中柱的个数 NumBin 是事先取定的，NumBin 越大，相似柱状图反映的子类中元素之间的相似性越精确。相似柱状图在区间[-1,1]上建立，每个柱的宽度 BinSize 固定，并且 BinSize=2/NumBin。相似柱状图中第 binId 个柱的长度 $H_i(\mathrm{bin})$ 表示子类 Sb_i 中相似值落在区间((binId-NumBin/2)×BinSize,(binId-NumBin/2))BinSize+BinSize)元素的对数。

相似柱状图构造过程可描述为：

输入：子类 Sb_i，柱的个数 NumBin，Sb_i 中元素的相似度矩阵 *SM*。

输出：具有 H_i 个柱的相似柱状图。

过程：令 $H_i(\mathrm{bin})=0$，bin=0,1,…,NumBin-1

对于子类 Sb_i 中的每一对元素 x 和 y，

Sim(x,y)=SM(x,y)；

If(Sim(x,y)=-1) then binId=0；

Else If(Sim(x,y)=1) then binId=NumBin-1；

Else binId=-1+⌈Sim(x,y)/BinSize⌉+(NumBin/2)；

++H_i(binId)；

End

Return H_i

End

图 1 是一个具有 20 个柱的子类的相似柱状图的例子。

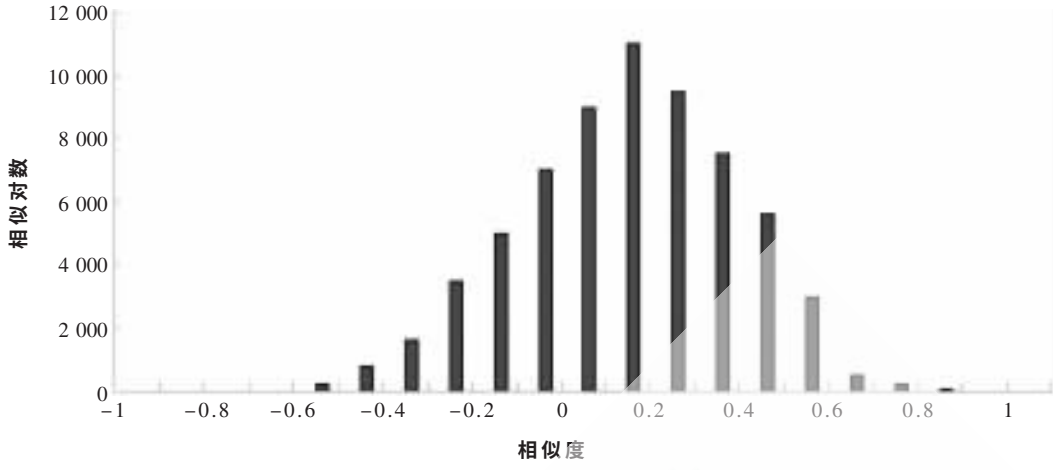

图 1 一个具有 20 个柱的子类的相似柱状图

4.3.2 融合因子

融合因子 $msf(Sb_i, Sb_j)$ 代表两个子类 Sb_i 和 Sb_j 的聚合性。子类之间的聚合性通过对应相似柱状图之间的聚合性来量化。这里设定各个子类的相似柱状图中柱的个数相等，H_i 和 H_j 聚合生成新的相似柱状图 H_{ij}。聚合公式为：

$$H_{ij}(\text{bin})=H_i(\text{bin})+H_j(\text{bin})+|\text{Sim}(x,y)|$$

$$(\text{bin}=0,1,\cdots,\text{NumBIn}-1;\text{并且}$$

$$((\text{binId}-\text{NumBin}/2)\times\text{BinSize})<\text{Sim}(x,y)$$

$$<((\text{binId}-\text{NumBin}/2)\times\text{BinSize}+\text{BinSize});) \quad (7)$$

$|Sb_i|$ 代表子类 Sb_i 的个数，$n_{\text{sim}}(Sb_i)=|Sb_i|\times(|Sb_i|-1)/2$ 代表子类 Sb_i 中元素之间两两相似的对数，$n_{\text{sim}}(Sb_i, Sb_j)=(|Sb_i|+|Sb_j|)\times(|Sb_i|+||Sb_j||-1)/2$ 代表子类 Sb_i、Sb_j 聚合后元素两两相似的对数。两个子类之间的融合因子的计算公式如下：

$$msf(Sb_i, Sb_j)=\frac{\sum_{\zeta}^{\text{numBin}-1}(((\text{bin}\times\text{binSize})-1+(\text{binSize}/2))\times H_{ij}(\text{bin}))}{n_{\text{sim}}(Sb_i, Sb_j)} \quad (8)$$

其中 ζ 是相似度阈值。

融合阶段的融合过程为：初始化每个子集为一个单独的类。从 *CMM* 矩阵中找到值最大的元素，也即子集中融合因子 msf 最大的两个子集，将其聚合成一个新的类，并且更新 *CMM* 矩阵。迭代以上过程，直到聚类个数为 L。

CBKM 算法的具体过程算法为：

输入：文本集合 X，二分聚类中的迭代次数 ITER，相似度阈值 ζ，期望得到的聚类个数。

输出：包含 L 个类的聚类 $S=\{S_0, S_1, \cdots, S_{L-1}\}$；

过程：(1)输入聚类的类的个数 L，$k=2$；

(2)用 KM 算法和 BKM 算法分别得到两个聚类集合 $S^{KM}(l)$、$S^{BKM}(l)$；

(3)利用聚类集合 $S^{KM}(l)$、$S^{BKM}(l)$ 得到合作应变矩阵 *CCM*，*CCM* 中的元素构成了一个包含 n_{sb} 个元素的新集合 Sb，这个集合中的元素不相交；

(4)对于每个 $Sb_i \in S_b$，建立对应的相似柱状图 $H(Sb_i)$；

(5)利用子类的相似柱状图，构建 *CMM* 矩阵，矩阵中第 i 行、j 列元素的值为 $msf(Sbi, Sbf)$，并初始化聚类集合 $S^{\text{cooperative}}(l_k)=Sb$；

(6)从聚类集合 $S^{\text{cooperative}}(k)$ 选取在 *CMM* 矩阵中值最大的两个元素合并，并更新 $S^{\text{cooperative}}(k)$。迭代此步骤直到聚类的个数为 L；

(7)返回聚类结果。

5 实验和算法评估

5.1 评估标准和测试数据集

任何一种聚类算法都要对聚类性能进行分析。本文对聚类结果采用 F 度量[5]的方法进行分析。F 度量是一种文本聚类的评测方法，它将信息检索技术中的准确率和召回率结合起来，其中，准确率和召回率分别用式(9)、式(10)表示。

$$\text{准确率 } precision(i,r)=n_{ir}/n_r \quad (9)$$

$$\text{召回率 } recall(i,r)=n_{ir}/n_r \quad (10)$$

式中，n_{ir} 是聚类 r 中包含类别 i 中的文本的个数，n_r 是聚类类别 r 中实际对象的数目，n_i 是原来预定义类别 i 应有的文本数。则聚类 r 和类别 i 之间 f 值计算式为：

$$f(i,r)=\frac{2recall(i,r)\times precision(i,r)}{recall(i,r)+precision(i,r)}$$

最终聚类结果的评价函数为：

$$F-measure=\sum_i \frac{n_i}{n}\max\{f(i,r)\}$$

其中 n 为测试文本的个数。$F-measure$ 值越高，聚类效果越好。

本实验采用的测试数据集是 Yahoo、UW、SN 和 20NG。这些数据集在进行测试之前要先进行分词处理，并且实现对于每一个数据集指定特定的聚类数进行聚类。

5.2 实验结果

本实验算法采用 Java 语言编写，选取的数据集包含

的中文本数目为 18 000~2 300。文本篇幅大小差异较大，长文本的字数有几万字，而短的只有几百字，具有一般性。针对不同的数据集人工指定不同的类数。实验结果如表 1 所示。

表 1 实验结果

F-measure	KM	BKM	CBKM
UW(k=10)	0.700	0.752	0.847
SN(k=17)	0.483	0.528	0.692
Yahoo(k=20)	0.459	0.550	0.678
20NG(k=20)	0.435	0.417	0.498

实验结果表明，CBKM 算法运用于 Yahoo、UW、SN 和 20NG 4 个测试数据集上的效果比较好。较之 KM 和 BKM 算法，CBKM 算法的 F-measure 值有一定的提高，聚类性较好。

参考文献

[1] 刘远超，王晓龙，徐志明，等.文本聚类综述[J].中文信息学报，2006，20(3)：55-62.

[2] SAVARESI S M，BOLEY D.On the performance of bisecting k-means and PDDP[C].Proceedings of the 1st SIAM International Conference on DataMining，2001：1-14.

[3] 张培颖，李村合.智能搜索引擎中个性化信息检索技术研究[J].科学技术与工程，2008，8(17)：5046-5049.

[4] HAMMOUDA K，KAMEL M.Collaborative document clustering[C].2006 SIAM Conference on Data Mining(SDM06)，2006：453-463.

[5] 索红光，王玉伟.一种用于文本聚类的改进 k-means 算法[J].山东大学学报，2008，43(1)：60-64.

(收稿日期：2010-01-29)

作者简介：

邹海，男，1969 年生，硕士生导师，教授，主要研究方向：中间件技术，信息检索。

李梅，女，1984年生，硕士研究生，主要研究方向：信息检索。