基于聚类特性的大规模文本聚类算法研究

The Research on a Large-Scale Text Clustering Algorithm based on Clustering Features

唐春生　金以慧

(清华大学自动化系　北京 100084)

Abstract Large-scale text processing becomes a great challenge as the fast growing of Internet and information explosion. Clustering is an effective method to solve this problem. An incremental algorithm called Mulit-Level CFK-means methods for large-scale text clustering is presented in this paper. More cluster information can be reserved and utilized by using the clustering features(CF) structure in this algorithm. Clustering results can be achieved very fast in one scan of the data. The computing and file exchange time of the algorithm is several times less than k-means algorithm and the accuracy of the results is almost equal to k-means algorithm. The effectiveness of the algorithm is proved by the contrastive experiment on Reuters text sets.

Keywords Clustering features(CF), Multi-level CFK-means algorithm, Text clustering

一、引言

随着 Internet 的飞速发展，人们能从网上得到更多的信息，但过多的信息常常会导致信息迷失。将信息进行分类是帮助信息利用的有效方法，聚类则是文本类别划分时常用的技术，其特点是不需训练集即可从给定的文本集合中找到聚类划分[1~5]。已有的聚类方法大多是针对小规模数据的，当计算资源和时间受到限制时，原有的大部分方法已不能满足要求，需要能够处理大规模数据的算法[6]。标准 k 均值方法是比较基本也是很常用的一种聚类方法，其计算复杂度与模式数量成线性关系，这使其具有处理大规模数据的可能。k 均值方法本质上是一种迭代的方法，当数据不能一次全部读入内存时，则需和磁盘进行多次数据交换，并且这种交换相应于迭代次数要反复多次，这无疑需要花费大量的 I/O 时间。

针对以上问题，本文提出一个适用于大规模数据的、递增的多层 CFK-means 聚类算法。算法的最大特点是只需要扫描全部数据一次，即可得到数据的聚类划分，聚类准确率和 k 均值方法基本持平，在有些文本聚类实验中甚至优于 k 均值算法；而其计算时间和文件交换时间比 k 均值算法少几倍。通过 Reuters 文本集数据进行的对比试验，证明了算法的有效性。

二、多层 CFK-means 算法的关键问题与实现

2.1 文本聚类问题的数学描述

在文本聚类问题中，文章通常用向量空间法来表示，即将文章中的词看成向量空间中的一维，于是文章可以表示成为一个词频向量，它是向量空间中的一个点[6]。

令 $\mathbf{X}=\{\mathbf{x}_1,\cdots,\mathbf{x}_n\}\subset R^d$ 为向量空间 R^d 中的一个有限文本集，其中第 i 个文本 $\mathbf{x}_i=\{x_{i1},\cdots,x_{id}\}$ 为一个 d 维向量，$\mathbf{X}$ 可以用一个 n×d 矩阵表示。

对文本集 $\mathbf{X}$ 进行聚类，是指通过评价文本之间的相似度，为每一个 $\mathbf{x}_i$ 分配一个类别标识 l_i，从而得到对应于文本集 $\mathbf{X}$ 的类别标识集 $L=\{l_1,\cdots,l_n\}$，其中 $l_i\in\{1,k\}$，k 为聚类数目。文本聚类的目标是使同一聚类中文本之间的相似度大于它与其他聚类中文本的相似度。

文章向量权重的计算一般采用 TFIDF 法，词频向量定义为：$x_{ij}=TF(w_i,x_j)IDF(w_i)$，其中 TF 指词频(即词 w_i 在文章 x_j 中出现的次数)，$IDF=\log\frac{D}{DF(w_i)}$，D 为文章总数，$DF(w_i)$为包含有词 w_i 的文章数[7]。文章向量一般要进行标准化处理，即使其长度为 1。

在文本聚类算法中，聚类常用其中心点来表示，

$$\bar{\mathbf{x}}=\frac{1}{N}\Sigma_{i=1}^{N}\mathbf{X}_i \tag{1}$$

文章向量之间的相似度一般用文章向量夹角的余弦来评价，当进行标准化处理后，计算公式简化如下：

$$c_{ij}=\cos(\mathbf{x}_i,\mathbf{x}_j)=\frac{(\mathbf{x}_i,\mathbf{x}_j)}{[(\mathbf{x}_i,\mathbf{x}_i)(\mathbf{x}_j,\mathbf{x}_j)]^{1/2}}$$
$$=(\mathbf{x}_i,\mathbf{x}_j)=\Sigma_{k=1}^{d}x_{ik}x_{jk} \tag{2}$$

其中 N 为聚类中文本的数量，$\mathbf{x}_i$，$\mathbf{x}_j$ 分别为两个 d 维的文章向量。

另外可以定义聚类的半径 R，它用聚类中各点到中心点的平均相似度的倒数来表示，即

$$\frac{1}{R}=\frac{1}{N}\Sigma_{i=1}^{N}\cos(\mathbf{x}_i,\bar{\mathbf{x}})=\frac{1}{N}\Sigma_{i=1}^{N}(\mathbf{x}_i,\bar{\mathbf{x}})$$
$$=(\frac{1}{N}\Sigma_{i=1}^{N}\mathbf{x}_i,\bar{\mathbf{x}})=(\bar{\mathbf{x}},\bar{\mathbf{x}})=\|\bar{\mathbf{x}}\|^2 \tag{3}$$

聚类半径可以作为聚类围绕中心点松紧度的衡量。若文本划分得比较正确，则聚类半径会比较小，反之则应当较大，根据这个准则可以指导文本的划分，选择使聚类半径增加最小的两类来合并。

用中心点来代表聚类可以使问题变得比较简洁，计算也相应简化，但这种方法会损失一些聚类信息，如聚类中模式的数量、聚类的松紧度等，因此在本文提出的大规模聚类算法中采用了聚类特性结构来表示聚类。

2.2 聚类特性的概念及其性质[8]

聚类特性(Clustering Features, CF)定义为一个三元组：CF=(N, LS, SS)

其中 N 为聚类中的数据点数目；

LS 为这 N 个数据点的线性加和，即 $\Sigma_{i=1}^{N}\mathbf{x}_i$，LS 仍为一个向量；

SS 为 N 个数据点的平方和，即 $\Sigma_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i$，SS 为一个数值。

利用聚类特性来表示聚类，可以保留更多的聚类信息，这对于提高聚类的质量有一定的作用，而聚类特性所具有的性质又保证了这种表示方法不会增加聚类过程的复杂度和计算

量，反而在一定程度上有助于两者的降低。

性质 1 设 A 和 B 为两个将合并的子聚类，其 CF 值分别为 $CF_A=(N_A,LS_A,SS_A)$ 和 $CF_B=(N_B,LS_B,SS_B)$，则合并后的聚类 C 的 CF 值为

$$CF_C=CF_A+CF_B=(N_A+N_B,LS_A+LS_B,SS_A+SS_B) \quad (4)$$

注意到一个数据点也可以作为一个子聚类，而聚类过程就是这些子聚类不断合并的过程，其中的计算可以利用性质 1 很简单地实现。

性质 2 在聚类过程中的一些参数，如中心点和聚类松紧度等可以由聚类特性直接计算得出。

$$\text{中心点：}\bar{x}=LS/N \quad (5)$$

$$\text{聚类半径：}R=\frac{1}{\left(\frac{1}{N^2}LS'LS\right)} \quad (6)$$

同理，如果定义其他参数和指标，也可以用聚类特性来表示。利用这种表示，可以减少很多的重复计算。

利用聚类特性的可加性和表示性，可以构造递增的、批处理的、适用于大规模数据的聚类算法。

2.3 多层 CFK-means 聚类算法的思想及算法描述

多层聚类方法的核心思想可以描述如下：根据内存资源的情况将全部数据切分成 p 批，每一批数据可以完全读入到内存中，然后将每一批数据进行标准 k 均值聚类得到 k 个聚类划分，如果用中心点来表示聚类，则全部数据处理完后可得到 pk 个聚类中心点，然后将这 pk 个中心点进行划分得到需要的 k 个聚类。如果数据量非常庞大，则上述过程可能需要进行多次。多层聚类方法的示意图如图 1 所示。

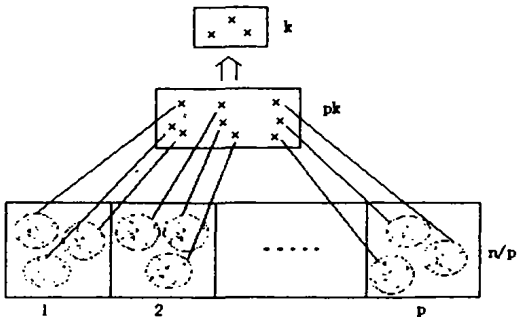

图 1 基于 CF 结构的多层聚类示意图

多层聚类方法的关键是找到一种能很好描述聚类信息的数据结构，并且这种数据结构能对聚类结果进行累加，而 CF 结构正是这样一种具有良好性质的数据结构。

基于 CF 结构的多层聚类方法采用了 CF 结构作为表示聚类的数据结构，它可以保留更多的聚类信息，这弥补了 k 均值方法只以中心点向量来表示聚类的不足；利用 CF 结构的累加性，可以将全部数据的聚类过程分解为多批部分数据的 k 聚类过程，而且批次聚类的结果可以进行累加和综合得到最终需要的聚类结果，这就实现了在一次扫描中完成聚类划分；利用 CF 结构的特点，算法可以有效利用已有计算所获得的信息，减少后续数据划分所需的迭代次数，极大地降低了计算复杂度和计算时间，提高了聚类的质量。

以下是多层 CFK-means 聚类算法的算法描述。

a)根据计算机内存资源状况开辟三个存储区，分别用来存放读入的批量数据、批次数据聚类后得到的中心点、暂时不进行聚类的奇异点；

b)选择 k 个初始中心点。中心点可以从数据点中随机选择，也可以在向量空间中随机生成 k 个点；

c)与数据文件进行交换，根据批量数据存储区的大小读入一批数据；

d)如果是第一批数据，则以 b)中选择的中心点作为初始中心点，否则以上一批次数据聚类结束后得到的聚类中心作为初始中心点，对当前批次的数据进行 k 均值聚类，直到收敛，此时得到的 k 个中心点存入到中心点存储区中；在聚类过程中，可以计算聚类半径等参数，以之作为数据划分的指导；当数据点超出一定的聚类半径时，可以将它们作为暂时奇异点，暂时不进行划分并将它们存储到奇异点存储区中；

e)修正中心点并更新中心点存储区内容，将批量数据存储区清空；

f)如果奇异点存储区已满，则对这些暂时奇异点进行一次 k 均值聚类，收敛后得到的中心点存到中心点存储区中，同时清空奇异点存储区；

g)如果中心点存储区已满，则将每个中心点看作一个数据点进行 k 均值聚类，但用它所包含数据点的个数来加权。得到的 k 个中心点取代原有的中心点，这是中心点压缩过程；

h)如果已经处理完所有数据点，则进入下一步，否则返回 c，进入下一次循环；

i)对中心点存储区中的中心点进行聚类，得到最终的聚类划分。

多层 CFK-means 算法是批量处理的方法，只需扫描全部数据一次即可得到聚类结果，所采用的 CF 结构使得所保存的聚类信息更丰富，而且这些信息是可以不断累积的，过程的递增性和结果的累加性使得新数据的划分可以在原有结果的基础上进行，不仅可以减少计算量，而且可以提高聚类的质量。

三、实验设计及结果分析

3.1 试验设计

为了检验多层 CFK-means 算法在文本聚类上的有效性，本文对多层 CFK-means 算法和标准 k 均值方法进行对比实验。对比实验主要从聚类准确率、计算时间、与数据文件交换时间、循环迭代次数等方面考察两种方法的差别。

为了统一基准和便于比较，各算法都采用批次读入数据的方法，而且都采用同样的初始中心点。由于 k 均值方法的结果与初始中心点的选择有关，而且和数据读入的顺序有关，因此实验中对每一组数据都生成 25 个经过数据随机排列的数据文件，对每一个文件进行随机采样得到 5 组不同的初始中心点，然后进行 5×25 次实验，从 5 组不同中心点的结果中选取最好的一个，然后对 25 个结果进行算术平均得到最终结果。

实验所采用的文本集合是 Reuters-21578 文章集[9]，使用“ModApte”切分，即利用其中 9603 篇作为训练集，3299 篇作为测试集。文集中包括 90 类，选择文章最多的 10 类进行测试。经过词切分和词频统计得到 10756 个词。

在实验中选取了 3×30，5×30，5×100，5×200 等四组数据构成(a)、(b)、(c)、(d)数据集合，其中 3×30 表示从文章最多的 10 类中选出 3 类，每类 30 篇文章组成数据集合。

3.2 实验结果

算法采用 Visual C++来实现，实验是在 k6 200，128M 内存的计算机上进行，表 1 比较了多层 CFK-means 算法和 k 均值算法的准确率。

表 1　多层 CFK-means 算法和 k 均值算法对比结果

数据集	样本数量	样本维数	标准类数	平均错分率(%)	
				k-means 算法	多层 CFK-means 算法
(a)	90	10756	3	23.7	24.4
(b)	150	10756	5	31.8	28.5
(c)	500	10756	5	33.5	25.8
(d)	1000	10756	5	32.6	25.5

图 2～4 分别为两种方法的平均计算时间图、平均文件交换时间图和平均迭代次数图。

3.3　结果分析

分析实验数据，得出了一些结果，并给出结论和原因分析，如表 2 所示。

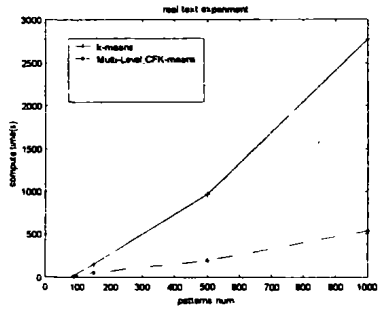

图 2　两种算法的平均计算时间

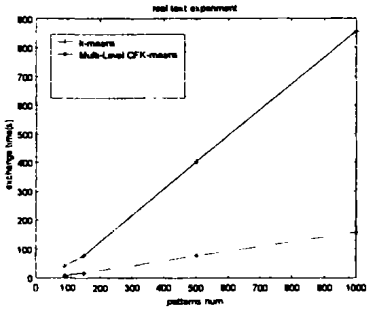

图 3　两种算法的平均文件交换时间

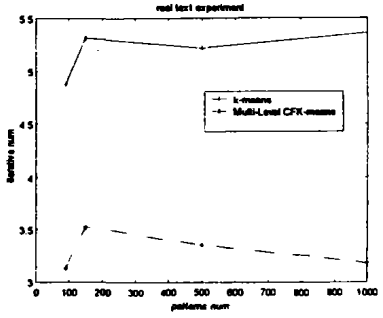

图 4　两种算法的平均迭代次数

表 2　实验结果及原因分析

实验结果	原因分析
1. 多层 CFK-means 方法是递增的方法，能在一次扫描中即得出聚类结果，而 k 均值方法需要多次扫描；	• 多层 CFK-means 方法所采用的 CF 结构具有累加性； • 多层 CFK-means 方法将全部数据划分分解为多批的部分数据聚类，其结果为批次聚类结果的综合；
2. 多层 CFK-means 方法的聚类正确率和 k 均值方法基本持平，在某些文本数据集上的实验甚至优于 k 均值方法；	• 多层 CFK-means 方法所采用的 CF 结构能较多地保留聚类信息，能够较好地指导后续数据的划分； • 多层 CFK-means 方法采用了聚类半径等指标来指导数据的划分，有助于减少错分； • 多层 CFK-means 方法采用的暂时奇异点技术有助于将一些边缘数据点正确划分； • 对于数据点相对较少而维数比较高的数据集，k 均值方法迭代得到的中心点可能比批次计算并不断综合得到的中心点准确度更差，这就影响了数据点的划分准确度；
3. 两种算法的平均计算时间和文件交换时间与数据集的模式数目基本成线性关系，但 k 均值算法在平均计算时间和平均文件交换时间上比多层 CFK-means 算法多出几倍、而且随着数据的增多这个比率也在增大；	• 两种算法所需的时间与模式数目、聚类数量、模式维数以及迭代次数有关，而多层 CFK-means 方法的迭代次数明显少于 k 均值方法； • 多层 CFK-means 方法所采用的 CF 结构将一些“乘法”运算转换为“加和”运算，其参数也可以通过 CF 来简单计算得出，不仅减少了运算所需的时间，而且有效地减少了重复运算； • 多层 CFK-means 方法能很好地利用已有数据划分所获得的信息，减少新数据划分时所需的计算；
4. 比较数据集(b)、(c)和(d)的结果，可以看出当数据增多时，多层 CFK-means 方法准确率有一些提高；	• 多层 CFK-means 方法在数据聚类的过程中能逐渐找到数据的分布特性，确定聚类的指标和中心，这使得后续数据的划分变成简单的归类，当数据点达到一定数量后，后续数据划分的准确率能够得到保证。

小结　文本是信息的重要载体之一，随着 Internet 的飞速发展，网上文本信息出现了爆炸趋势，这使得文本信息处理成为一个急需解决的问题。文本聚类是不需训练集即可划分出类别的一种方法，它能有效地解决文本的自动划分问题。经典的聚类方法一般都是针对小规模数据的情况，对于大规模的文本聚类问题，原有方法已不能满足要求。

针对上述问题，本文提出了一种基于 CF 结构的适用于大规模数据的多层 CFK-means 算法，这种算法可以在一次扫描后即得出聚类结果，而且聚类效果和 k 均值方法相当，从实验结果看出，该算法的聚类准确率能达到 70%左右，而所需的计算时间和文件交换时间少于 k 均值方法数倍，这在文本数据聚类实验中得到了验证。在实际文本聚类应用中，可以先用简单的关键词检索进行粗略划分，然后用 CFK-means 方法对划分结果进行聚类，这样可以进一步提高准确率；另外，对文章向量进行降维处理，在减少计算量的同时也可以提高准确率。

本文提出的算法可以很方便地实现并行化处理，可以将各批次的数据交给各 CPU 进行计算，计算完毕后进行一次同步，将这一轮得到的中心点进行综合得到下一轮数据聚类的初始中心点。同样，本文的算法也可以做分布式处理，这将在今后的工作中做进一步的研究。

参 考 文 献

1　Yang Y. An evaluation of statistical approaches to text categorization. Journal of Information Retrieval, 1999, 1(1/2): 67～88
2　Jain A K, Farrokhnia F. Unsupervised texture segmentation using Gabor filters. Pattern Recognition, 1991, 24(13): 1167～1186
3　Anderberg M R. Cluster analysis for applications. New York, NY: Academic Press, Inc., 1973
4　Bjorner L, Chinatsu A. Fast and effective text mining using linear-time document clustering. In: KDD-99, San Diego, California, 1999
5　Salton G. Developments in automatic text retrieval. Science, 1991, 253: 974～980
6　Jain A K, Murty M N, Flynn P J. Data clustering: A review. ACM Computing Surveys, 1999, 31(3): 264～323
7　Salton G, et al. A vector space model for automatic indexing. Communications of the ACM, 1975, 18: 613～620
8　Zhang T, Rughu R, Miron L. BIRCH: an efficient data clustering method for very large databases. In: Proc. of the ACM SIGMOD Intl. Conf. on Management of Data, ACM, 1996. 103～114
9　http://www.research.att.com/～lewis/reuters21578.html