一种基于频繁概念集的文本聚类方法①

A Text Clustering Method Based on Frequent Concept-Sets

肖 杰 黄汉永 张 驹 (中南大学 信息科学与工程学院 湖南 长沙 410083)

摘 要: 针对传统文本表示模型的不足以及文本向量的“高维诅咒”问题，本文提出一种基于频繁概念集的文本聚类方法(CFC)。该方法利用 HowNet 将文本中的关键词映射为概念，然后使用 Apriori 算法找出概念文本集中的频繁特征项，我们称之为频繁概念，最后利用 CFC 算法实现文本聚类。实验表明，较传统的基于频繁特征项的同类方法，该方法能获得更好的聚类效果。

关键词: 文本聚类 概念映射 频繁项集 CFC 算法

1 引言

文本聚类分析是组织文档的一种有效方式。许多传统聚类技术虽然也应用于文本聚类，但很难满足文本聚类的特殊需求：数据的高维度，可理解的类别标签，聚类个数预先不可知，文本间潜在的语义关系，允许簇重叠。

文本聚类的首要工作是文本表示。传统文本聚类算法大多采用基于关键词的文本表示模型，该模型只考虑词的频率信息，忽略了词与词之间的语义信息，影响了聚类质量。同时，文本聚类是一种典型的基于高维空间的数据挖掘任务，直接在这种高维空间进行聚类，将导致聚类性能和聚类质量大大下降。此外，将结果簇个数作为算法的输入参数，以及对文档集进行硬划分的做法也是不合理的。

针对以上问题，本文提出一种基于频繁概念集的文本聚类方法(Clustering based on Frequent Concept-Sets，CFC)。CFC 算法使用 HowNet 将关键词映射为同义词集合所代表的概念[1]，然后采用 Apriori 算法挖掘出频繁特征项集，我们称之为频繁概念集。基于这些频繁概念集，利用一种新的聚类方法实现对原始文本集的聚类。通过融合本体知识和频繁特征项，该算法有效降低了特征空间的维数，提高了聚类准确性。实验表明，CFC 算法较传统基于频繁特征项的同类方法能获得更好的聚类效果。

2 概念映射

传统文本聚类方法采用基于关键词的文本表示模型表示文本，文本之间的相似度计算基于关键词。然而，由于自然语言的复杂性，同一词意可通过多个不同的词形来表达，传统的文本表示模型无法发现隐藏在文本中的这种语义信息。

假设文本集 $D=\{d_1,d_2\}$，d_1={计算机 微机}，d_2={电脑 computer}。由于文本 d_1，d_2 没有共享任何关键词，如果采用基于关键词的方法进行聚类，同主题的两个文本将被分到不同的簇中，从而降低了聚类准确度。如果把关键词{计算机,微机,电脑,computer}映射为同一概念"电脑"，就可以使同主题的文本划分到同一簇中。

可见，基于概念的聚类不仅能发现隐藏的语义信息，还能降低特征空间的维数。进行概念映射前，本文先进行特征词筛选，降低特征空间的维数，获取主题区分能力较强的关键词。

2.1 特征词筛选

算法首先对文本进行分词，去除停用词，获得一个特征空间缩减的文本集。由于文本集中主题词比例较小，大多数词出现频率很低，这些低频词不仅增加了计算复杂度，还淹没了主题，因此，本文利用词语的频率信息对特征词进行筛选。我们选择前个最大的词作下一步的概念映射，其中为文本集中非重复词的个数，根据经验，取 0.1 时聚类质量不会受到太大影

① 收稿时间:2008-10-15

响[2]。该步骤去除了大部分噪声词，保留了主题区分能力较强的词，大大降低了特征空间的维数。

2.2 概念映射

文献[3,4]采用 WordNet 本体来改善聚类质量，但 WordNet 不能直接用于中文文本聚类，本文采用 HowNet 来获取词语所代表的概念。在 HowNet 中，每个词都用义项 DEF 来定义，本文假定 DEF 义项为该词语的概念。由于一个词可能对应多个义项，概念映射就是要准确地获取词语在 HowNet 中所代表的概念。

对于文本中的单义词并且在 HowNet 中存在的词，直接获取该词语的一个相应概念。对于有多个义项的词，本文认为该词的某个义项在原文中出现次数越多，对词的支持度就越大，从而被认为是该词的真实义项，也即该词的实际概念。对于未登录词，将词本身作为概念。

经过概念映射，原文本中频率较低而与文本同主题的关键词的重要程度得到了提升，整个文本的主题更加突出。通过概念映射，文本的关键词集合转化为概念集合。

3 CFC算法

“高维诅咒”是文本挖掘面临的普遍问题，基于频繁项集的聚类技术试图解决这个问题。最早的频繁集聚类算法是由 Wang 等人提出的 FTC 算法，该算法产生平面簇；Beil 等人提出层次状聚类算法 HFTC 算法，但聚类结果依赖项集的选择顺序。Benjamin[5]提出一种基于频繁项集的层次聚类算法 FIHC 算法，该算法先把包含某个频繁词集的文本全部归入该簇，再使用一个打分函数去除簇之间的重叠，较前期算法具有更好的聚类质量。

但以上聚类算法都是基于频繁关键词集的，没有考虑文本间的语义关系。此外，在初始簇分离过程中，对具有相同文本集而不同频繁项个数构成的初始簇，FIHC 算法往往将这些文本集划分到频繁项数目较少的初始簇中，这与基于频繁项集聚类算法中关于频繁项数目越多，对簇的描述能力越强的原则是相背的。CFC 算法和传统的同类算法一样，也是基于频繁项集的，不同的是该算法中的项指的是概念，而不是传统意义上的关键词。算法克服了传统同类方法存在的缺点，并实现了文本集的软聚类，提高了聚类质量。

3.1 基本定义

定义 1(频繁概念集).将文本集 D_c 看成数据库，D_c 中的文本看成数据库中的事务，D_c 中所有概念的集合就是数据库中项的集合。设 C 为概念的集合，X 是 C 的非空子集，$percent(X)$ 表示包含集合 X 的文本占总文本的比例，如果 $percent(X) > min\_sup$，则 X 称为一个频繁概念集。

其中，min_sup 为频繁概念集的最小支持度，其值可由用户指定，也可让算法在抽样文本集上进行多次预运行获取一个最合适的值。

定义 2(簇重叠).设 T_A 为与支持簇 A 的文本的集合，T_B 为支持簇 B 的文本的集合，且 $T_A \neq T_B$，如果 $T_A \cap T_B \neq \varnothing$，簇 A 与簇 B 存在重叠现象，称为簇重叠。

定义 3(文本对簇的支持度).设有簇 C 和文本 d，如果文本 d 包含了簇中的所有频繁概念，则称文本 d 支持簇 C，文本支持的簇的个数称为文本对簇的支持度，用 $cluster\_sup$ 表示。

定义 4(k 阶簇).包含 k 个频繁概念的簇

定义 5(非重叠度).设 C_{ik} 为第 i 个簇，其阶为 k，$|C_{ikj}|$ 为文本集第 j 趟归属处理后支持该簇的文本个数，N 为文本集 D 中的文本总数，$\alpha = N \Big/ \sum_{i=1}^{m} |C_{ikj}|$ 称为非重叠度，其中 m 为初始簇的个数。

从定义 5 可知，α 的取值范围为(0,1]，α 越大，重叠度越低。对于 FIHC[4]算法，其最终 α 值为 1，即不存在簇重叠现象。

3.2 初始簇构建与软分离

利用 Apriori 算法可以挖掘出基于概念的频繁项集，也即频繁概念集。把每个频繁概念集看成一个初始簇，所有包含了该频繁概念集的文本被划到该初始簇中。由于初始簇中的每个文本都包含了该频繁概念集，可以把该概念集看成对该簇的类别描述。

由于一个文本可能包含多个频繁概念集，初始簇之间将出现簇重叠现象，本文利用新的算法决定文本的归属簇。同时，由于多主题文本的存在， 新算法试图实现初始簇的软分离，即多主题文本可考虑归到多个相关簇。此外，用户可选择性地指定结果簇的个数，未指定时算法将自动生成结果簇。

本文算法主要包括两部分：第一部分基于以下两个原则实现部分文本的唯一簇归属。

(1) 支持 k 阶频繁概念集的文本同时也支持该频繁集的所有子集

(2) 由于初始簇是由其频繁概念的集合来描述的，我们认为一个簇中的频繁概念越多，对簇的描述能力就越强

算法从最大同阶初始簇开始处理，对于同阶重叠初始簇中的非重叠文本，将它们从所有的低阶初始簇中删除，实现部分文本的唯一簇归属。

第二部分对其余重叠文本进行簇归属处理，实现簇的软分离。算法循环处理每一个重叠的文本，将其归到一个或多个簇中。每趟循环执行之前，计算簇的非重叠度 α，并与一个非重叠度阈值 θ 进行比较，如果 $\alpha > \theta$，则算法结束，否则对重叠文本实施簇归属。θ 值可由用户指定，也可取默认值，实验表明当 θ =58%时获得最佳聚类效果。如果用户指定了结果簇个数，在每趟循环执行之前，将当前得到的非空簇个数与用户指定的结果簇个数进行比较，如果相等则算法结束。在对重叠文本实施簇归属时，算法仍从最大同阶簇开始处理。根据上述原则(2)，总是将重叠的文本优先归到最大 k 阶簇。对于同阶的重叠簇，则计算各个簇内频繁概念的权值和，将重叠的文本归到权值和最大的簇中。若存在多个并列最大权值簇，则将重叠文本归到各个簇。

CFC 算法通过设置重叠度阈值 θ，实现了初始簇的软分离，而 θ 的大小则决定了初始簇的分离程度，即结果簇的重叠程度。

3.3 CFC 算法

设 max 为所有初始簇的阶的最大值，对于 k 阶簇，$k \in \{1,2,\ldots,max\}$；T_{ik} 为支持第 i 个簇的文本集合，阶为 k；T_{ks} 表示所有支持 k 阶簇的文本中支持度为 s 的文本集合；可选参数 noc 表示结果簇的个数。

算法 1 CFC 算法描述：

输入：初始簇，阈值 θ，可选参数 noc

输出：经过软分离的结果簇

步骤：

① 令 k=max，序列 j=1

② 对所有小于 k 阶的初始簇，从支持该簇的所有文本中删除那些出现在集合 T_{ik1} 中的文本

③ max=max-1，如果 max>1，则执行②，否则重置 max，令 k=max，执行⑥

④ 计算当前非空的簇的个数 n，如果 n=noc，则转到步骤⑧，否则计算簇的非重叠度 α，如果 $\alpha < \theta$，则 j=j+1，执行步骤⑥，否则转到步骤⑧

⑤ k=max

⑥ 对 $T_{ks}(s \geq 2)$ 中的文本 d_j，计算 d_j 支持的簇中各频繁概念在中的权值和，将重叠文本 d_j 归到权值和最大的簇，若存在多个并列最大簇，则将 d_j 归到各个最大簇，并将 d_j 依次从它支持的其它簇对应的文本集合中删除。如果 $T_{ks} \neq \varnothing$，则执行步骤④，否则转到步骤⑦

⑦ max=max-1，如果 max>1，则返回步骤⑤，否则转到步骤⑧

⑧ 选择 $T_{ks} \neq \varnothing$ 的簇作为结果簇，算法结束。

4 实验和结果分析

4.1 评测数据

实验使用 2 个中文语料作为评测数据。第一个语料选自 Sougou 实验室的文本分类语料，该语料库来源于 Sohu 新闻网站保存的大量经过手工整理与分类的新闻语料与对应的分类信息。其分类体系包括几十个分类节点，网页规模约为十万篇文档。我们从精简版语料中随机选择 1000 篇文本作为评测数据，它们分别来自 IT，健康，体育，教育，军事 5 个类别，每个类别 200 篇文本，并将该语料标记为 Sougou-TC。

另一个语料是从互联网上搜集的 1440 篇文本。根据网上的原始分类体系，将这些文本采用手工形式粗略地划分为 55 个主题，其中最大的主题包括 80 篇文本，最小的包括 4 篇文本，并将该语料记为 CFC-55。

4.2 评测方法

本文采用常用的评价指标 F 值[6]评价聚类质量，这是一种标准的外部评测方法，可用于评测平面和层次聚类结果。它将每个聚类结果簇视为一个查询的结果，每个预定义类别视为一个查询的相关文档集合。对于每个人工类和结果簇，准确率、召回率和 F 值的计算如下：

$$Recall(K_i, C_j) = \frac{n_{ij}}{|K_i|} \quad (1)$$

$$Precision(K_i, C_j) = \frac{n_{ij}}{|C_j|} \quad (2)$$

$$F(K_i, C_j) = \frac{2 * Recall(K_i, C_j) * Precision(K_i, C_j)}{Recall(K_i, C_j) + Precision(K_i, C_j)} \quad (3)$$

其中，n_{ij} 表示簇 C_j 中包含预定义类别 K_i 中的文本个数。对每一个预定义类别 K_i，找出一个最能描述它的

结果簇，即使 $F(K_i,C_j)$ 最大的 C_j。对一个聚类结果 C，通过取它对所有预定义类别的最大 F 值的加权平均来衡量其聚类质量，该指标称为聚类结果 C 的总体 F 值，记为 $F(C)$：

$$F(C)=\sum_{K_i\in K}\frac{|K_i|}{|D|}\max_{C_i\in C}\{F(K_i,C_j)\} \quad (4)$$

其中，K 表示所有预定义类别，C 表示所有结果簇；$|K_i|$ 表示类 K_i 中文档数目，$|D|$ 表示数据集 D 中全体文档数目，$F(C)$ 取值范围是[0,1]，$F(C)$ 值越大表明聚类效果越好。

4.3 评测结果

为了评测 CFC 算法的聚类质量，我们选取 BKM 和 FIHC 两种聚类方法与本文算法进行横向比较。首先为 3 种算法指定聚类结果簇的个数，使其与各语料的预定义类别数相同。然后将三种方法在同一数据集上分别执行 10 次，取平均 F 值进行比较。BKM 算法随机选择初始聚类中心，CFC 算法的非重叠度阈值取 0.6。为公平起见，FIHC 和 CFC 算法取相同的最小支持度，在这里取三种支持度 2%,5%和 8%。

表 1 列出了 3 种聚类方法在两种语料上不同支持度下聚类结果的 F 值。

表 1　BKM，FIHC 和 CFC 方法的 F 值对比

语料名称	最小支持度	F值		
		BKM	FIHC	CFC
Sougou-TC	2%	0.352	0.476	0.501
	5%	0.401	0.508	0.513
	8%	0.375	0.512	0.498
CFC-55	2%	0.395	0.493	0.524
	5%	0.408	0.547	0.517
	8%	0.422	0.584	0.601

从表 1 中可以看出，本文算法在两种数据集上取不同的最小支持度时，较其它两种聚类方法的聚类质量都有明显的改善。我们认为这种改善来自两个方面，一是 CFC 算法是一种基于概念的聚类方法，另一方面 CFC 算法允许结果簇的重叠。因此，CFC 算法是一种聚类效果较好，实用性较强的聚类方法。

接下来测试 CFC 算法对参数的敏感度。在不指定结果簇个数的情况下，计算了 10 个不同非重叠度阈值下获得的 F 值，测试结果如图 1 所示。当 $\theta=1$ 时，CFC 算法实现的是一种硬聚类，每个文本被归到唯一的簇中，结果簇没有重叠。由图 1 可以看出，当 θ 的值小于 0.6 时，聚类质量受 θ 的影响很小。

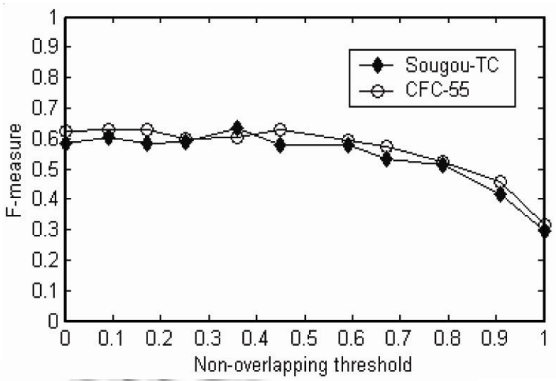

图 1　非重叠度阈值对 CFC 算法 F 值的影响

5　结论

本文通过分析文本聚类的特殊需求和前期算法的不足，提出一种基于频繁概念集的聚类方法。该方法将结果簇个数作为可选的输入参数，提高了算法的灵活性。通过参数控制实现了初始簇的软分离，使聚类结果更具合理性。算法融合了语义本体和频繁项集聚类思想，可以进一步降低特征空间的维度，有效改善传统以词为特征的聚类算法的质量，不足之处是对知识库具有较大依赖性。

参考文献

1 刘远超,王晓龙,徐志明,关毅.文档聚类综述.中文信息学报, 2006,20(30):58－59.

2 Liu YC, Wang XL, Wu C. ConSOM: A conceptional self-organizing map model for text. Clustering Neurocomputing, 2008(71):857－862.

3 Hotho A, Staab S, Stumme G. Ontologies improve text document clustering. Proceedings of the 3rd IEEE International Conference on Data Mining, 2003:541－544.

4 Li YJ, Chung SM, Holt J. Text Document Clustering Based on Frequent Word Meaning Sequences. Data and Knowledge Engineering, 2008, 64(1):381－404.

5 Fung BCM, Wang K, Ester M. Hierarchical document clustering using frequent itemsets. Proceedings of SIAM International Conference on Data Mining, 2003.

6 Bellare M, Rogaway P. The game-playing technique. Cryptology ePrint Archive Report. 2004. http://eprint.iacr.org/.