基于语义列表的中文文本聚类算法 *

马素琴, 施化吉, 李星毅
(江苏大学 计算机科学与通信工程学院, 江苏 镇江 212013)

摘 要: 针对大多数基于向量空间模型的中文文本聚类算法存在高维稀疏、忽略词语之间的语义联系、缺少聚簇描述等问题,提出基于语义列表的中文文本聚类算法 CTCAUSL(Chinese text clustering algorithm using semantic list)。该算法采用语义列表表示文本,一个文本的语义列表中的词是该文本中出现的词,从而降低了数据维数,且不存在稀疏问题;同时利用词语间的相似度计算解决了同义词近义词的问题;最后用语义列表对聚簇进行描述,增加了聚类结果的可读性。实验结果表明,CTCAUSL 算法在处理大量文本数据方面具有较好的性能,并能明显提高中文文本聚类的准确性。

关键词: 文本聚类; 文本表示; 语义列表; 相似度计算; 聚簇表示
中图分类号: TP311 **文献标志码**: A **文章编号**: 1001-3695(2010)05-1697-03
doi:10.3969/j.issn.1001-3695.2010.05.024

Chinese text clustering algorithm using semantic list

MA Su-qin, SHI Hua-ji, LI Xing-yi
(*School of Computer Science & Telecommunication Engineering, Jiangsu University, Zhenjiang Jiangsu* 212013, *China*)

Abstract: Common Chinese document clustering algorithms rely on the so-called vector space models, to solve the problems in these methods, such as the text characteristic of high dimensions and sparse space, ignoring the semantic relations among words, and lack of the description of cluster, this paper proposed a Chinese text clustering algorithm using semantic list(CTCAUSL). The algorithm used documents as semantic lists. Words in a document semantic list were those existing in this document, so reduced dimensions and there was no sparse space. In the meantime, the method used the similarity calculation to solve the synonym or near-synonym problem. Then, in order to improve the readability of cluster results, described clusters by semantic lists. The experimental results indicate that CTCAUSL performs well in dealing with a large number of document data, and has significantly improved the accuracy of Chinese text clustering.

Key words: text clustering; text representation; semantic list; similarity calculation; cluster representation

0 引言

随着 Internet 以及各种文本管理系统的发展,文本数据资源越来越丰富,如何从大量的文本数据中发现潜在有价值的文本信息,进而对文本数据进行有效管理、分析和利用就愈来愈引起关注。文本挖掘、信息过滤和信息检索等已成为新的研究热点,而快速高质量的文本聚类技术则是组织文本的关键技术之一,它能挖掘语料的潜在结构,将文本划分成有意义的子簇,使同簇中的文本尽可能相似,而不同簇中的文本差异尽可能大[1],从而协助人们更好地对大规模文本进行理解,同时也可作为一种有效的预处理步骤,为进一步的文本分析提供初步的语料结构。

传统的文本聚类算法大致可以分为基于划分的方法和基于层次的方法,它们大多数以向量空间模型[2](VSM)为基础。该表示方法非常简单,但存在高维稀疏的问题[3];而且,基于 VSM 的聚类算法没有很好地解决文本数据所特有的两个自然语言问题,即近义词和同义词;同时对聚类后的簇没有提供可以理解的描述。这些问题影响了文本聚类算法的效率和准确性,使得文本聚类的性能下降。为此许多学者提出改进方法[4,5],文献[4]利用文本的具体语义来计算文本间的相似度,提高了聚类的精度;文献[5]提出一种语义内积空间模型,从而导出语义、词和文本相似度的计算方法,基于该模型的聚类方法比基于向量空间模型的算法聚类性能更好。但是文献[4,5]都没有解决高维稀疏的问题,也没有对聚类结果进行描述。另外一些基于语义相似度的文本聚类算法,如文献[6]仅将文本表示成一个名词列表,列表中的名词互不相同。以这种文本表示法为基础的聚类算法效果并不好,其重要原因是该方法忽略了单词的词频对文本间语义相似度的影响。也有学者对其进行了改进,提出概念列表表示法[7],该方法有效地解决了词频对文本间语义相似度的影响,但没有考虑同义词和近义词在同一个文本中同时出现的问题。

为此,本文提出一种基于语义列表的中文文本聚类算法 CTCAUSL。该算法采用语义列表表示文本,一个文本的语义列表中的词是该文本中出现的词,相对于向量空间模型中的词是所有文本集中的词,数据维数降低了很多,而且不存在稀疏

收稿日期: 2009-09-12; **修回日期**: 2009-10-21 **基金项目**: 国家自然科学基金资助项目(60841003);国家火炬计划资助项目(2004EB33006)

作者简介:马素琴(1980-),女,河南周口人,硕士研究生,主要研究方向为数据挖掘(makexin2000@163.com);施化吉(1964-),男,教授,博士研究生,主要研究方向为数据挖掘、计算机网络与分布计算、企业应用集成;李星毅(1969-),男,副教授,博士,主要研究方向为数据挖掘、空间数据库、交通信息系统和控制理论.

问题;同时语义列表中有一项是指向其同义词或近义词的指针,在计算词间相似度时,首先比较两个词及其同义词和近义词,如果这两个词是同义词(或近义词),可直接得出它们之间的相似度是1.00(或是0.98),提高了词间相似度计算的效率及准确性;最后对聚簇进行语义列表表示,增加了聚类结果的可读性。

1 基于语义列表的中文文本聚类算法

1.1 语义列表表示法

文本表示成如下形式:

$$D=\{(W_1,f_1,P_1),(W_2,f_2,P_2),\cdots,(W_n,f_n,P_n)\} \tag{1}$$

其中:W_i 表示在一个文本中出现的词;f_i 为 W_i 在文本中出现的次数;P_i 是指向 W_i 的同义词和近义词的指针,该项主要解决同义词近义词的问题。由于 W_i 是在一个文本中出现的词,维数比向量空间模型表示法降低了很多,也不存在稀疏的问题,称这种文本表示法为语义列表表示法。由此可见,语义列表就是一个由词、词频和指针组成的三元组列表,语义列表中的各个词互不相同。该方法不仅解决了高维稀疏的问题,还有效地处理了同义词和近义词的问题。

1.2 聚簇的语义列表表示

称聚类后所得到的簇为聚簇。已有的文本聚类算法对结果没有描述,不利于人们对聚类结果的查看和了解。为此,本文把一个聚簇中的所有文本视为一个大文本,于是聚簇也可以用语义列表表示。一个元组(W_i,f_i,P_i)中,W_i 是聚簇中的关键词;f_i 为词 W_i 在聚簇中出现的次数除以聚簇中文本的个数,因描述聚簇与文本表示不同,在这里要稍作改动;设置 P_i 为空。

1.3 相似度计算

1.3.1 词之间相似度的计算

在国内已有不少学者对中文词语相似度作了研究。本文使用基于《知网》的词语之间相似度的计算。

《知网》中有两种重要的概念:“概念”与“义原”。“概念”是对词汇语义的一种描述,每一个词可以表达为几个概念。“概念”是用一种知识表示语言来描述的,这种知识表示语言所用的词汇叫做“义原”。“义原”是用于描述一个“概念”的最小意义单位。所以要计算语义之间的相似度,首先要计算义原之间的相似度。相关定义如下:

定义1 义原深度。指义原 p 在整体义原层次体系中所处的层数位置,记为 $\mathrm{depth}(p)$。

定义2 重合度。指两个义原 p_1 和 p_2 在义原层次体系中所拥有的相同父节点的路径长度,记为 $\mathrm{spd}(p_1,p_2)$。

定义3 相异度。指两个义原和在义原层次体系中沿父节点逐步上移,直到两者达到第一个共同节点所走过的最短路径长度,记为 $\mathrm{dsd}(p_1,p_2)$。相异度与语义距离等价。

结合义原深度、重合度和相异度对义原的不同度量方式,定义义原相似度计算公式如下:

$$\mathrm{sim}(p_1,p_2)=\frac{2\times\mathrm{spd}(p_1,p_2)}{\mathrm{dsd}(p_1,p_2)+2\times\mathrm{spd}(p_1,p_2)}=\frac{2\times\mathrm{spd}(p_1,p_2)}{\mathrm{depth}(p_1)+\mathrm{depth}(p_2)} \tag{2}$$

对于《知网》而言,词分为实词和虚词两类。其中,实词和虚词差别很大,可直接令实词和虚词的语义相似度为0。对于虚词而言,因为《知网》总是用{句法义原}或{关系义原}进行描述,所以只需计算去掉花括弧后的句法义原或关系义原的相似度即可。

在《知网》中,实词语义可分成四个部分:a)第一基本义原描述式,DEF项中的第一个义原;b)其他基本义原描述式,DEF项中除第一义原外的其他独立义原或具体词;c)关系义原描述式,DEF项中用“关系义原=基本义原”或“关系义原=(具体词)”描述语义的部分;d)符号义原描述式,DEF项中用“关系符号基本义原”或者“关系符号(具体词)”描述语义的部分。此处把任意两个语义 C_1 和 C_2 各部分的相似度分别记为 $\mathrm{sim}_1(C_1,C_2)$、$\mathrm{sim}_2(C_1,C_2)$、$\mathrm{sim}_3(C_1,C_2)$、$\mathrm{sim}_4(C_1,C_2)$。语义 C_1 和 C_2 的整体相似度为

$$\mathrm{sim}(C_1,C_2)=\beta_1\mathrm{sim}_1(C_1,C_2)+\sum_{i=2}^{4}\beta_1\beta_i\mathrm{sim}_i(C_1,C_2) \tag{3}$$

其中:$\beta_i(1\leqslant i\leqslant 4)$是一个可调节的参数,各部分的重要程度通过 β_i 进行限定,并满足 $\beta_1+\beta_2+\beta_3+\beta_4=1$,$\beta_1\geqslant\beta_2\geqslant\beta_3\geqslant\beta_4>0$。式(3)中 β_1 与 β_2 相乘的意义在于,语义中主要义原起了决定性作用,其相似度将对次要部分的相似度起较强的制约作用,其他三部分则相对独立。

1.3.2 文本之间相似度的计算

一个文本经过预处理后得到一个语义列表,文本的含义是由语义列表中词的联合含义表示。所以,两个文本间的相似度可以通过它们所包含的词的语义相关程度来计算。本文将文本间的语义相似度定义如下:

$$\mathrm{sim}(d_1,d_2)=\mathrm{sim}(X_1\wedge\cdots\wedge X_m,Y_1\wedge\cdots Y_n)=\frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n}w_i\times w_j\times\mathrm{sim}(X_i,Y_j) \tag{4}$$

其中:X_i 和 Y_i 分别为文本 d_1 和 d_2 语义列表中的词;w_i 是词 X_i 在文本 d_1 中出现的权重,w_j 是词 Y_j 在文本 d_2 中出现的权重。n 的定义如下:

$$n=\sum_{i=1}^{m}\sum_{j=1}^{n}w_i\times w_j \tag{5}$$

用 n 进行标准化,减少了因为文本语义列表中词的个数过多而造成文本相似度增大的问题。

w_i 的定义如下:

$$w_i=f_i\times\log(N/n+1) \tag{6}$$

其中:f_i 是词 W_i 在 d_i 文本中出现的次数;N 是整个文本集中文本的个数;n 是词 W_i 在文本集中出现过的文本的个数。

1.4 基于语义列表的中文文本聚类算法

1.4.1 CTCAUSL算法

CTCAUSL算法的主要思想是通过语义列表来表示中文文本,一个文本语义列表中的词是该文本中出现过的词,而一个文本的VSM表示方法中的词是文本集中所有文本中出现过的词,前者与后者相比维数大大降低了;另一方面,语义列表中有一个指向词的同义词或近义词的指针,这在计算词语间的相似度时大大节省了时间,也处理了同义词和近义词的问题。为验证语义列表表示法的性能,本文使用聚类算法中最常用的分裂层次聚类方法进行聚类;最后对聚簇进行语义列表表示,增加了聚簇的可读性。所以,基于语义列表的中文文本聚类算法,从理论上讲可以有效地提高中文文本的聚类性能。

CTCAUSL算法结合中文文本的特点,采用语义列表表示中文文本,用文本间的语义相似度作为文本间相关程度的度量。由式(4)(5)可知,相似度矩阵是一个对称矩阵,而且如果两个文本中有两个词的语义相似度不为零,则这两个文本的相似度也不为零。在聚类之前,相似度矩阵代表一个连通图。为

此,本文根据文本表示的特点以及连通图的性质,使用分裂(自上而下)的层次聚类算法以减少算法的时间复杂度。在每次分裂中,将不满足阈值的矩阵元素设一个标志,表示相连的两个节点不再相互连接。重构矩阵的连通分量,如果连通分量的个数大于等于输入聚类个数 K,则停止循环,否则继续分裂。分裂停止后,为解决一个簇中不相邻接的节点不一定相似的问题,在各个簇对应的连通图中求得一个包含最多节点的完全图,以保证簇中各个节点必是相似的。最后,计算各个非聚类节点与各个簇的相似度,将其归入与之最相近的簇中。非聚类节点与簇间相似度的定义如下:

$$\mathrm{sim}(d,C)=\frac{1}{|C|}\sum_{d_i\in C}\mathrm{sim}(d,d_i) \tag{7}$$

其中:d 是非聚类节点;C 为簇;$|C|$ 表示簇中节点的个数。

CTCAUSL 算法描述如下:

输入:相似度矩阵 M,聚类的个数 K。
输出:K 个聚类。

```
begin
P = componNumber( G );//componNumber( G )
/ * 采用深度优先遍历,获得连同分量的个数 * /
while( P > = K)
 {
计算各个连通分量的最小相似度阈值:{T1,…,Tp};
最小相似度阈值 T = min( {T1,…,Tp} )
将所有不符合阈值 T 的矩阵元素设置一个标志位,表示这两个节点不再连接;
P = componNumber( G );
 }
C = M 中各个连通分量中,节点个数最多的完全图形成聚簇;
while( 存在不属于任何聚簇中的节点 d)
 {
  计算节点 d 与各个聚簇的相似度;
  将 d 归入与之最相似的聚簇中;
 }
end
```

本算法采用 K 百分比相似度计算相似度最小阈值,定义如下:

$$\mathrm{sim\_threshold}=\mathrm{sim}-\mathrm{max}-k(\mathrm{sim\_max}-\mathrm{sim\_min}) \tag{8}$$

其中:K 为输入参数,满足 $0<K<1$,sim_max 为最大相似度值,sim_min 为最小相似度值。

1.4.2　聚簇描述

可以直接用语义列表作为聚簇描述,本文选择语义列表中的一部分词作为聚簇描述。选择的规则如下:

a)在聚簇中出现的次数越多,越容易被选择为聚簇描述。

b)词的信息量越大,越容易被选为聚簇描述。

因此,本文将语义列表中词的权值定义为

$$\mathrm{CW}(w)=(f)^{\alpha}\times[\mathrm{IC}(w)]^{\beta} \tag{9}$$

其中:f 是词 W 在聚簇中出现的次数(是语义列表中词 W 对应元组中的 f);$\alpha+\beta=1$,分别定义了词出现次数和信息内容对权值的影响程度。当定义聚簇描述时,根据用户要求取权值最大的 n 个词作为聚簇描述。

2　实验

2.1　实验环境与实验数据

为验证 CTCAUSL 算法的聚类性能,实验在如下环境下完成:Windows XP 操作系统,2.11 GHz 的 CPU,1 GB 的内存,160 GB 的硬盘,编程工具为VC ++。使用“基于《知网》的词汇语义相似度计算”软件包,该软件包是基于《知网》2000 版完成的。软件使用简单的对话框界面,如图 1 所示。

图 1　词语相似度计算界面

为检验 CTCAUSL 算法的准确性和扩展性,对 CTCAUSL 算法和基于 VSM 的聚类算法的性能进行了实验比较。实验使用搜狐研发中心搜狗实验室的文本分类语料库[8],实验准备了四类主题的中文文本,即汽车、财经、体育和军事。使用以 VSM 为基础的层次(由上而下)聚类算法与 CTCAUSL 算法进行了四组对比实验,聚类簇数设定为四类。第一组实验中每类文本选 40 篇;第二组实验中每类文本选 80 篇;第三组实验中每类文本选 120 篇;第四组中每类文本选 160 篇。

2.2　实验步骤

实验分以下五个步骤:a)文本预处理。先用中国科学院计算机所软件室开发的 ICTCLAS 分词工具[9]对文本数据进行分词。为提高聚类的准确性,本文对分词后的结果进行了停用词过滤、词性过滤和无效词条过滤。停用词指的是一些出现频率很高但没有实际意义的词,如“是、的、所、到、可、能、由”等一些功能词;词性过滤指的是分词后根据分词结果的词性标注信息,只保留名词、动词和缩略词这些实词。实验证明,形容词和副词这些实词的引入并没有使聚类结果有所改善,却增加了特征的维数和聚类的时间;无效词条过滤中,无效词条包括高频词和低频词,对于聚类而言,这些词大多是无用信息。最后对处理过的词进行特征选择[10]。b)采用语义列表表示文本。在文本表示之前,先建立一个数据字典,这个数据字典中的词是文本集中出现的所有词,并根据词频从大到小的顺序位于字典中,然后根据《知网》计算出各个词的同义词和近义词,并存储这些信息;最后根据数据字典用语义列表表示文本,这样不但节省存储空间还解决了同义词和近义词记词的问题。c)先用基于《知网》的软件计算词语间的相似度,最后算出文本间的相似度。d)采用两种方法进行聚类并输出结果。e)对输出簇进行语义列表表示。

2.3　实验结果与分析

准确率又称为精度或正确率,主要用来衡量聚类结果中某个簇本身的凝聚程度。本文采用的聚类结果度量标准是被正确分类的文本个数占所有输入文本个数的比例,其计算公式为 $p=N/n$。其中:n 是输入文本的个数;N 是被正确分类的文本个数。显然,聚类正确率越高,其聚类的性能也就越好。实验结果如图 2 所示。

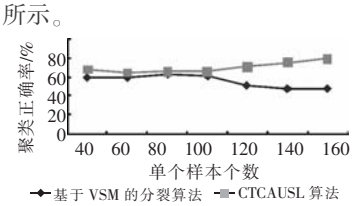

图 2　两种算法的聚类结果比较

基于 VSM 聚类方法的最大缺点就是忽略了词与词之间的语义信息、各维度之间的联系,导致文本的相似度计算不够精确。本文利用《知网》提供的丰富的语义信息,从语义上具体分析文本内容,通过语义距离计算文本间相似度。CTCAUSL 算法中的最小相似度阈值是各个连通分量阈值的

(下转第 1707 页)

从图 3 中可以看出,EASI 算法未能成功将源信号从混合信号中分离出来,可见对超高斯和亚高斯信号混合的情况,EASI 算法在性能上有很大的下降。从图 4 中可以看出,KDMEICA 算法成功地分离出了源信号。图 5 是在 2 000 次迭代过程中 *PI* 性能指数的变化曲线。可以看出,无论是选用 tanh(10y)还是 y^3,EASI 算法 *PI* 值均不能收敛于 0,所以难以成功地分离出源信号。相反,KDMEICA 算法迭代 650 次收敛,*PI* 值趋近于 0。由于 KDMEICA 算法在每一次迭代过程中准确地估算信号的概率密度分布,并由此不断更新非线性函数,在杂系信号混合的情况下,算法具有很好的分离性能和鲁棒性。

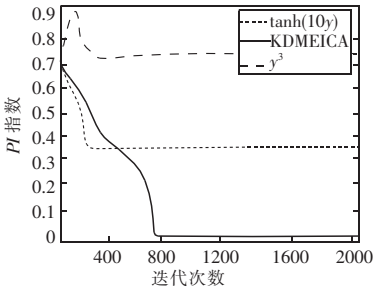

图 5　*PI* 性能比较

4　结束语

本文提出了一种基于核密度最大熵的非线性函数更新算法,在算法学习过程中,实时地估计分离信号的概率密度函数,由此得出非线性评价函数并应用到盲分离方法,给出了相应的算法。实验证明,基于核密度最大熵的盲分离算法能有效地从杂系混合信号中分离出源信号,实现了“全盲”地处理混合信号。

参考文献:

[1] LEE T W, GIROLAMI M, SEJNOWSKI T J. Independent component analysis using an extended infomax algorithm for sub-Gaussian and super-Gaussian sources[J]. Neural Computation, 1999, 11(2): 409-433.

[2] MATHIS H, DOUGLAS S C. On the existence of universal nonlinearities for blind source separation[J]. IEEE Trans on Signal Processing, 2002, 50(5): 1007- 1016.

[3] ALIBRANDI U, RICCIARDI G. Efficient evaluation of the pdf of a random variable through the kernel density maximum entropy approach [J]. International Journal for Numerical Methods in Engineering, 2008, 75(13): 1511-1548.

[4] MEAD L R, PAPANICOLAOU N. Maximum entropy in the problem of moments[J]. Journal of Mathematical Physics, 1984, 25(8): 2404-2417.

[5] CARDOSO J R, LAHELD B. Equivariant adaptive source separation [J]. IEEE Trans on Signal Processing, 1996, 44(12): 3017-3030.

[6] YANG H H, AMARI S. Adaptive online learning algorithms for blind separation: maximum entropy and minimum mutual information[J]. Neural Computation, 1997, 9(7): 1457-1482.

[7] BOSCOLO R, PAN Hong, ROYCHOWDHURY V P. Independent component analysis based on nonparametric density estimation [J]. IEEE Trans on Neural Networks, 2004, 15(1): 55-65.

[8] CARDOSO J. Infomax and maximum likelihood for blind source separation[J]. IEEE Signal Processing letters, 1997, 4(4): 112-114.

(上接第 1699 页)最小值,保证了每次分裂都发生在文本相似度相对较小的聚簇中。为了保证聚簇中各个文本相互之间都是相似的,将算法求得连通分量中的全连通图作为聚簇,再将不属于任何聚簇的节点根据相似度将其归入与之最相似的聚簇中,这就解决了因为传递性造成非相似的文本被归入一类的问题。实验结果表明,CTCAUSL 算法在聚类正确率上有明显提高;另外,CTCAUSL 算法采用了语义列表表示中文文本,一个文本的语义列表中只保存该文本出现过的词,实验过程中发现文本表示的维数大大减少了,且不存在稀疏问题。由图 2 可明显看出,采用 CTCAUSL 算法的准确率要高于基于 VSM 的层次聚类算法,同时 CTCAUSL 算法表现出较好的扩展性。

3　结束语

本文提出一种基于语义列表的中文文本聚类算法 CTCAUSL。该算法用语义列表表示文本,能够有效地解决传统聚类算法中存在的高维稀疏问题,同时也解决了同义词近义词的问题,而且对聚类结果进行了描述,增加了聚类结果的可读性。实验结果表明,CTCAUSL 算法是一种有效的中文文本聚类算法。

参考文献:

[1] 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.

[2] SALTON G, WONG A, YANG C S. A vector space model for automatic indexing[J]. Communication of the ACM, 1975, 18(5): 613-620.

[3] SONG Wei, PARK S C. A novel document clustering model based on latent semantic analysis[C]//Proc of the 3rd International Conference on Semantics, Knowledge and Grid. Washington DC: IEEE Computer Society, 2007: 539-542.

[4] 冯少荣,肖文俊. 基于语义距离的高效文本聚类算法[J]. 华南理工大学学报,2008,36(5):30-37.

[5] 彭京,杨冬青,唐世渭,等. 一种基于语义内积空间模型的文本聚类算法[J]. 计算机学报,2007,30(8):1354-1363.

[6] PANDYA A, BHATTACHARYA P. Text similarity measurement using concept representation of texts[C]//Proc of the 1st International Conference on Pattern Recognition and Machine Intelligence. Berlin: Springer, 2005: 678-689.

[7] SUN Shuang, ZHANG Yong. Clustering method based on semantic similarity[J]. Journal of Nanjing University of Aeronautics and Astronautics, 2006, 38(6): 712-716.

[8] 搜狗实验室语料库[DB/OL]. [2009-03-12]. http://www. sogou. com /labs/resources. html.

[9] LEEHAO-BUPT. ICTCLAS 中文分词工具[EB/OL]. [2008-12-05]. http://download. csdn. net/source/377228.

[10] LI Min-qiang, ZHANG Liang. Multinomial mixture model with feature selection for text clustering[EB/OL]. (2008). http://www. elsevier. com/locate/knosys.