一种结合词项语义信息和 TF-IDF 方法的文本相似度量方法

黄承慧[1),2)] 印 鉴[1)] 侯 昉[2)]

[1)](中山大学信息科学与技术学院 广州 510006)

[2)](广东金融学院计算机科学与技术系 广州 510520)

摘 要 传统的文本相似度量方法大多采用 TF-IDF 方法把文本建模为词频向量,利用余弦相似度量等方法计算文本之间的相似度.这些方法忽略了文本中词项的语义信息.改进的基于语义的文本相似度量方法在传统词频向量中扩充了语义相似的词项,进一步增加了文本表示向量的维度,但不能很好地反映两篇文本之间的相似程度.文中在 TF-IDF 模型基础上分析文本中重要词汇的语义信息,提出了一种新的文本相似度量方法.该方法首先应用自然语言处理技术对文本进行预处理,然后利用 TF-IDF 方法寻找文本中具有较高 TF-IDF 值的重要词项.借助外部词典分析词项之间的语义相似度,结合该文提出的词项相似度加权树以及文本语义相似度定义计算两篇文本之间的相似度.最后利用文本相似度在基准文本数据集合上进行聚类实验.实验结果表明文中提出的方法在基于 F-度量值标准上优于 TF-IDF 以及另一种基于词项语义相似性的方法.

关键词 文本聚类;词项语义相似度;文本相似度;自然语言处理

中图法分类号 TP311 **DOI 号**: 10.3724/SP.J.1016.2011.00856

A Text Similarity Measurement Combining Word Semantic Information with TF-IDF Method

HUANG Cheng-Hui[1),2)] YIN Jian[1)] HOU Fang[2)]

[1)](*School of Information Science and Technology, SUN Yat-Sen University, Guangzhou* 510006)

[2)](*Department of Computer Science and Technology, Guangdong University of Finance, Guangzhou* 510520)

Abstract Traditional text similarity measurements use TF-IDF method to model text documents as term frequency vectors, and compute similarity between text documents by using cosine similarity. These methods ignore semantic information of text documents, and semantic information enhanced methods distinguish between text documents poorly because extended vectors with semantic similar terms aggravate the curse of dimensionality. This paper proposes a similarity measurement, which is based on TF-IDF method, and analyzes similarity between important terms in text documents. This approach uses NLP technology to pre-process text, and uses TF-IDF method to filter those key terms that have higher TF-IDF value than other common terms. With the proposed data structure TSWT (Term Similarity Weight Tree) and the definition of semantic similarity, this paper resolves the semantic information of those key terms to compute similarities between text documents. Finally, several K-Means clustering methods is used for evaluating performance of the new text document similarity. By comparing with TF-IDF and another the-state-

收稿日期:2010-07-05;最终修改稿收到日期:2011-03-17.本课题得到国家自然科学基金(61033010)、国家科技重大专项基金(2008ZX10005-013)、广东省科技计划项目(2009A080207005,2009B090300450,2010A040303004)资助.**黄承慧**,男,1976 年生,博士研究生,讲师,主要研究方向为数据挖掘、自然语言处理、语义网.E-mail: hch.gduf@163.com. **印 鉴**(通信作者),男,1968 年生,博士,教授,博士生导师,主要研究领域为数据挖掘、机器学习、数据仓库.E-mail: issjyin@mail.sysu.edu.cn. **侯 昉**,男,1975 年生,博士,讲师,主要研究方向为海量数据存储、人工智能、计算机应用.

of-art semantic information based similarity method, experimental results on benchmark corpus demonstrate that it can promote the evaluation metrics of F-Measure.

Keywords　text clustering; term semantic similarity; text similarity; natural language process

1　引　言

文本聚类是指自动地将文本集合分组为不同的类别,同一个类别中的文本之间是非常相似的,而不同类别之间的文本则不相似[1].文本聚类过程中有几个关键问题:如何度量两个文本之间的相似性?如何确定文本聚类的数目?以及如何评价聚类是否自然地反映了文本自身的属性?在这些问题当中,如何建立文本之间相似性的度量是文本聚类的一个核心问题.

文本的相似度量是一个在语言学、心理学和信息理论等领域内被广泛研究的一个重要话题.传统的文本相似度量方法大都将文本看作一组词的集合体,分析每个词在文本中出现的次数以及在整个文本集合中出现的次数,进而利用这些词频信息将文本建模为一个向量,并利用向量间的余弦相似度、Jaccard 相似度等方法计算文本之间的相似度[2].基于语义的文本相似度量方法则通过同义词、冗余和蕴涵等语义关系来考察文本之间的相似性[3-4].

文本相似度量方法在许多领域有着广泛的应用:在信息检索领域,文本相似度量方法被认为是改进检索效果最好的方法之一[5];在图像检索领域,利用图像周围的文本可以获得更好的检索精度[6];此外,文本相似度量方法还广泛地应用于文本分类[7]、文本摘要的自动生成[8]、文本的重复检测[9]等领域.

基于词频向量的相似度方法忽略了文本中词项的含义,也忽略了文本中的语法、组织结构等信息.此外,对于大多数文本数据库而言,词项的数目和文本数目通常都很大,而采用词频向量模型,必须将文本表示为词项数目与文本数目大致相当的矩阵,矩阵中的行列向量都有着非常高的维度并且是极度稀疏的,最终导致了非常低效的计算[10].基于词项语义来考察文本相似度量的方法在文本表示模型上多数沿用了词频向量模型,没有针对文本表示的高维模型进行降维处理,也缺乏衡量文档之间相似程度的定义,导致基于词项语义信息的文本相似度量方法局限于一些特定的应用领域.

本文针对上述方法存在的缺陷,提出了一种既能有效降低文本表示模型的维度,又能结合词项语义信息进行相似度量计算的方法.给定两个文本,通过本文提出的算法能够高效、自动地计算出两个文本在语义层次上的相似度,并且能够在较为广泛的应用领域内使用.

2　相关工作

TF-IDF 方法是文本相似度量的方法中最为典型的一种.该方法基于下面的经验观察,将文本表示为文中出现的 n 个加权词项组成的向量[2]:

(1)词频(Term Frequency).某个词项在一个文本中出现的次数越多,它和文本的主题越相关;要注意在特定的语言环境下都有许多特定的词不具备这种特性而应将其排除,如中文的"的""地"、英文的"a""an"等.

(2)逆文本频率(Inverse Document Frequency).某个词项在文本集合的多篇文本中出现次数越多,该词项的区分能力越差.例如:在一个包含 1000 篇文本的集合中,如果某个词项 A 在 100 篇文本中都出现,而另一个词项 B 只在 10 篇文本中出现,则词项 B 比 A 具有更好的区分能力.

利用上述概念计算每一个词项 w_i 的 TF-IDF 值,通常采用如下公式:

$$TF\text{-}IDF(w_i)=tf(w_i)\times idf(w_i)$$
$$=tf_j(w_i)\times\log(N/df(w_i)) \quad (1)$$

式(1)中的 $tf_j(w_i)$ 表示当前词项 w_i 在文本 j 中出现的频率,N 表示文本集合中所有文本的总数,$df(w_i)$ 表示文本集合中有多少篇文本出现了当前词项 w_i.通过对文本集合中的每一个词项都进行上述分析,得到每一篇文本中每一个词项的 TF-IDF 值.然后再利用这些 TF-IDF 值为每一篇文本建立一个向量模型,通过计算向量间的余弦相似度或者 Jaccard 系数来确定文本之间的相似性.

随着互联网的发展,如何从海量的文本数据中获取更为准确的信息对这种忽略词项语义的方法提出了挑战.我们必须能够更加精确地分析、捕捉和刻

画文本的含义而不仅仅是词项出现的频率.例如一篇关于银行(bank)的文章和一篇关于河岸(bank)的文章,由于银行和河岸两者的词项都是 bank,基于词频的相似度量方法就很可能会将它们看成是很相似的文章.而一篇关于苹果和一篇关于橘子的文章则可能因为两者的词项不同(apple 和 orange)而认为是不相似的两篇文章.

基于上述观察,人们开始研究词和词之间的相似度.词与词之间的相似度量需要将所有的词组织起来构成一个词义的网络,通过考察该网络中词与词之间的边、节点等信息来建立词与词之间的相似度.最常用的是普林斯顿大学研究开发的 WordNet.文献[11]考察了词义网中密度、节点深度、链接类型等因素提出了一种基于词义网边的词与词之间的相似度量方法.文献[12]则给出了既考虑节点信息内容又结合节点之间边的方法.上述文献主要对名词或动词之间的相似度应用词典所构建的词与词之间的层次关系进行研究,对于形容词、副词而言,组织一个类似于名词的层次关系是非常困难的.文献[13]利用 WordNet 研究了局部相关性信息以此来确定文本之间的相似性.文献[14]则在文本词汇分布满足特定概率模型的前提下应用信息论的原理定义了词与词之间的相似度.

文献[3]基于上述词与词相似度量的思想,同时应用词义消歧方法改进传统的基于词频的文本相似度量,效果比传统的基于词频的方法有一定的提高,然而该方法没有降低文本模型的维度.文献[15]基于文本句子聚类的技术提出了一种判定句子相似性的方法,并将该方法应用于文本自动摘要中.文献[16]提出使用本体对搜索引擎返回的结果重新计算文本的相关性,并重新排序,在计算的过程中需要和用户进行交互以得到更高的检索效果.文献[4]则应用词项相似度量方法结合 WordNet 来改进文本的向量表示模型,通过分析文本中的概念、同义词和词项的上下位关系将原有的词频向量改进为用同义词,上下位关系词等更为广泛意义的词频向量,进而通过计算向量间的余弦相似度来进行文本聚类.这些方法没有降低文本表示向量的维度,文本之间相似度的计算方法也是传统的向量间余弦相似度.

基于上述对文本相似性方法的分析,本文提出了一种在传统 TF-IDF 方法表示文本的基础上结合词项语义计算文本之间相似度的方法.本文的贡献有以下几个方面:首先应用 TF-IDF 方法选取文本中的重要词项,有效地降低文本模型的维度,为文本的语义相似度计算提供一个合适的表征模型.其次,通过分析重要词项的语义相似性,给出了文本相似度的定义.此外,由于两篇文本中较高相似度的词项对文本的相似性计算比相似度较低的词项更有指示意义,为此本文提出了一种根据词项相似度大小对文本相似度计算进行加权处理的词项相似度加权树,用于指导文本相似度的计算.最后通过几种主流的聚类实验,验证本文提出的文本相似度量方法是否有效.实验对比了传统的 TF-IDF 相似度量方法和文献[4]提出的语义相似度量方法.实验表明,我们的方法在 F-度量指标上优于这两种方法.

3 基于词项语义的文本相似度

3.1 文本预处理

尽管原始的文本包含最完备的文本信息,然而目前的自然语言处理技术无法完全处理这些文本信息.因此,在对文本建立词项的词频向量之前,对文本进行适当的预处理是有必要的.传统文本的预处理主要是删除文本中对应于停用词列表中的特定词项,如中文的“的”“地”、英文的“a”“an”等.由于本文提出的方法需要对词项进行语义分析,除了删除停用词外还需要进行下面两个预处理步骤.

首先需要处理文本中的人名、地名、组织机构名称等特殊词项.一方面,在 TF-IDF 的计算中,这些特殊词项通常都会具有较高的 TF-IDF 值,从而容易导致对文本关键词项的错误选择.另一方面,文本中人名、地名、组织机构名称等特殊词项在进行词项之间的相似度计算时也会产生较大的影响,也需要对文本中的这些特殊词项进行区分.本文采用了命名实体识别技术来处理文本中的人名、地名、组织机构名称等特殊词项,将这些识别之后的特殊词项统一替换为特定的字符串.人名、地名、组织机构名称统一替换为 PER、LOC、ORG 等.在进行 TF-IDF 值较高的关键词项的选择时,可以忽略这些词项,避免了其对文本聚类的影响.其次,最能表征文本含义的主要是文本中的实词.因此,必须对文本中的词项进行词性分析,给出词项的语义属性,即该词项是名词、动词、形容词还是副词等.

3.2 关键词项选择

文本预处理完成后,需要对整个文本集合中每

一篇文本的词项进行 TF-IDF 值的计算,并将文本中各个词项的 TF-IDF 值表示为一个向量,以此进行文本的相似度计算. 这个文本向量是高维而且极度稀疏的. 根据信息论,IDF 的值实际上是一个特定条件下词项概率分布的交叉熵,而 TF 则是用来增加词项的权重,以便更好地描述文本中词项的信息特征. 因此,我们可以从每一篇文本中挑选若干重要的词项,以此来表征文本. 这样就能够做到在保证不影响文本特征提取的前提下,最大可能地减少文本特征向量表示的维度. 具体的做法是:把每一篇文本中词项的 TF-IDF 值进行排序,从中选取 TF-IDF 值大于 p(p 为百分比)的名词和动词词项作为关键词项,以此关键词项向量作为文本的特征表示. 与传统的 TF-IDF 方法相比,一篇文本的关键词向量维度下降了 $1-p$,这在效率上是一个较大的提高.

3.3　文本相似度的计算

得到了每一篇文本的关键词向量,接下来要考虑的就是如何计算两篇文本之间的相似度. 由于关键词项代表了一篇文本中最重要的信息,因此文本的相似度就可以由关键词项向量间的相似度来描述. 因此,文本之间的相似度就转换为关键词项向量间的相似度. 此外,由于每一篇文本的长度不尽相同,因而表征每一篇文本的关键词项向量的维度也不一样,我们必须消除这些影响,使得关键词项向量间的相似度满足基本的相似度量标准.

文献[17]给出了两个对象间相似度定义所要满足的基本条件. 如果 $Sim(x,y)$ 是数据点 x 和 y 之间的相似度,应满足以下条件:

当且仅当 $x=y$ 时,$Sim(x,y)=1(0\leq Sim(x,y)\leq 1)$.

对于所有的 x 和 y,$Sim(x,y)=Sim(y,x)$.

关键词向量可以表征文本,因此我们用关键词向量之间的相似度作为文本的相似度.

设 $\boldsymbol{v}_i,\boldsymbol{v}_j$ 是两篇不同文本的关键词项向量. 其中 $\boldsymbol{v}_i=(w_{i1},w_{i2},w_{i3},\cdots,w_{im})$,$\boldsymbol{v}_j=(w_{j1},w_{j2},w_{j3},\cdots,w_{jn})$. 定义文本相似度为

$$TextSim(\boldsymbol{v}_i,\boldsymbol{v}_j)=wf\times VectSim(\boldsymbol{v}_i,\boldsymbol{v}_j) \quad (2)$$

上式中 wf 表示关键词向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 之间相似度的加权因子,$VectSim(\boldsymbol{v}_i,\boldsymbol{v}_j)$ 表示关键词向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 之间的相似度. 如果两篇文本中彼此相似度较高的词项越多,而词项所占的 TF-IDF 值在各自文档中比例越高,说明这些词项更能反映它们在文本中的重要性,因此我们根据关键词向量中满足相似度阈值条件的关键词项的 TF-IDF 值在整篇文本 TF-IDF 值总和中所占的比例进行加权,具体的加权因子计算公式由式(3)给出.

$$wf=1+ave(i,j)\times(\sqrt{VectSim(\boldsymbol{v}_i,\boldsymbol{v}_j)}-VectSim(\boldsymbol{v}_i,\boldsymbol{v}_j)) \quad (3)$$

$$ave(i,j)=\frac{1}{2}\left(\frac{\sum\limits_{k\in\Lambda_i}TFIDF(w_{ik})}{\sum\limits_{k=1}^{m}TFIDF(w_{ik})}+\frac{\sum\limits_{l\in\Lambda_j}TFIDF(w_{jl})}{\sum\limits_{l=1}^{n}TFIDF(w_{jl})}\right) \quad (4)$$

式(4)中 $TFIDF(w_{ik})$ 表示关键词项 w_{jk} 的 TF-IDF 值,右端项表示关键词向量 $\boldsymbol{v}_i$ 中所有满足相似度阈值条件的关键词项 $w_{ik}(k\in\Lambda_i)$ 的 TF-IDF 值在 $\boldsymbol{v}_i$ 所有的词项 TF-IDF 值总和中所占的百分比. 式(4)中的集合 Λ_i 和 Λ_j 定义如下:

$$\Lambda_i=\{k:1\leq k\leq m,\max_{1\leq l\leq n}\{Sim(w_{ik},w_{jl})\}\geq\mu\},$$
$$\Lambda_j=\{l:1\leq l\leq n,\max_{1\leq k\leq m}\{Sim(w_{jl},w_{ik})\}\geq\mu\} \quad (5)$$

如果关键词向量 $\boldsymbol{v}_i$ 中的某个关键词 w_{ik} 与另一个关键词向量 $\boldsymbol{v}_j$ 中的关键词 $w_{jl}(l=1,2,\cdots,n)$ 的相似度超过用户设定的相似度阈值 μ,则将该关键词 w_{ik} 放入集合 Λ_i. 集合 Λ_j 包含的元素依据集合 Λ_i 的方法对关键词向量 $\boldsymbol{v}_j$ 中的关键词进行选择. $Sim(w_{jl},w_{ik})$ 表示关键词 w_{jl},w_{ik} 之间的语义相似度.

$$VectSim(\boldsymbol{v}_i,\boldsymbol{v}_j)=\frac{1}{2}\left(\frac{1}{m}\sum_{k=1}^{m}\max_{1\leq l\leq n}\{Sim(w_{ik},w_{jl})\}+\frac{1}{n}\sum_{l=1}^{n}\max_{1\leq k\leq m}\{Sim(w_{ik},w_{jl})\}\right) \quad (6)$$

$VectSim(\boldsymbol{v}_i,\boldsymbol{v}_j)$ 由向量 $\boldsymbol{v}_i,\boldsymbol{v}_j$ 中所包含的词项相似度决定,相似的向量必定包含相似度较高的词项,而不相似的向量则彼此所包含的词项相似度较低. 为了更好地根据词项相似度进行文本相似度的计算,本文在计算文本相似度时设计了一个词项相似度加权树的数据结构 TSWT(Term Similarity Weight Tree),该数据结构用于计算加权因子 wf. TSWT 是一个高度为 3 的平衡树,它包含叶结点和非叶结点,词项之间相似度超过某个阈值 μ 的词项按照相似度从大到小的顺序组织成一个有序队列保存在叶结点中;而非叶结点则保存词项最大、最小、平均相似度以及词项数目等集合信息. 图 1 给出了 TSWT 的结构.

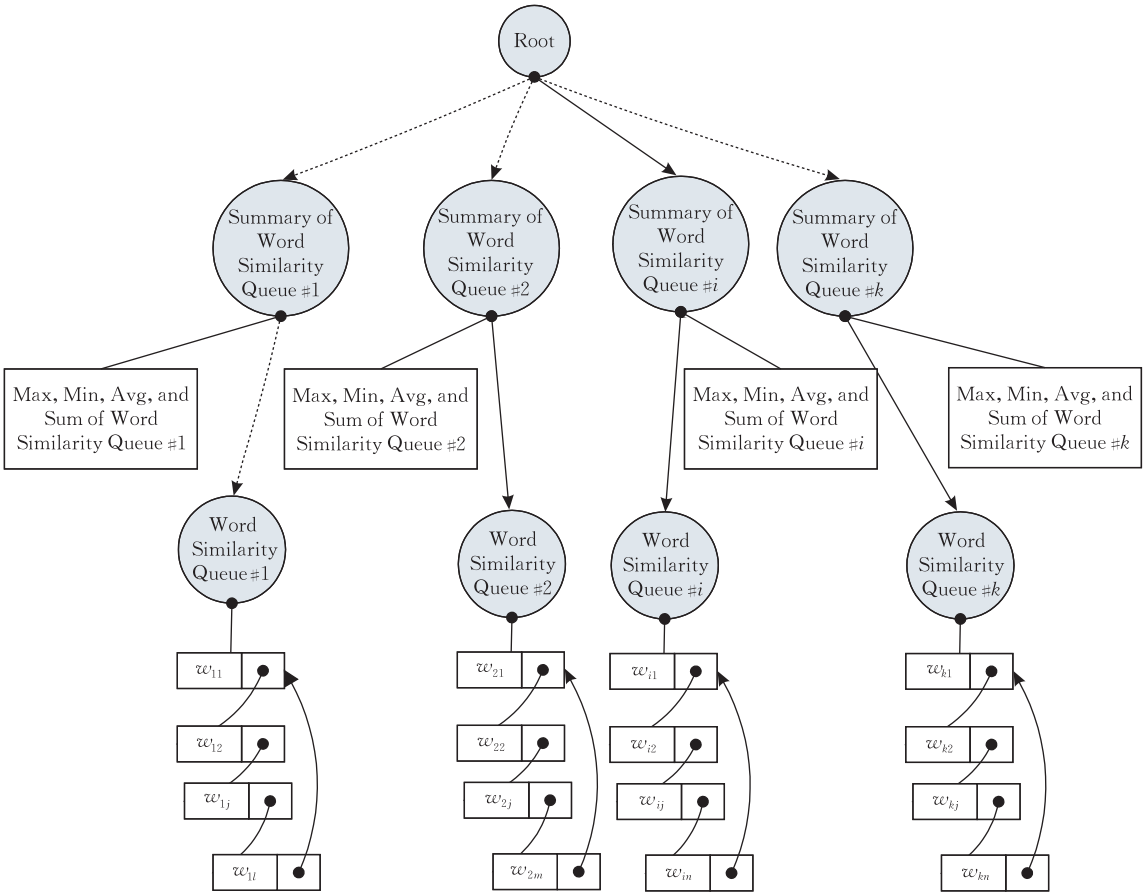


图 1 词项相似度加权树 TSWT

能够根据下面的方法，自动地构建词项相似度加权树 TSWT：

(1) 初始化.

TSWT 由用户根据特定任务选择特定的关键词项进行构建(如果用户没有指定初始的关键词项，随着关键词向量相似性的计算，根据相应的更新规则，系统能够自动构建相应的词项相似度加权树). 树中每个叶节点所包含的关键词项之间的相似度都超过某个阈值 μ，并且根据词项与当前文本类别其他词项的相似度从高到低将关键词项组织为有序队列.

(2) TSWT 加权及其更新.

在进行关键词项向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 之间的相似度计算过程中，如果关键词向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 中的某一对关键词项 w_{ik}，w_{jl} 满足以下条件之一则对关键词项向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 计算的相似度结果进行加权处理. 加权的权重由满足相似度阈值 μ 条件的关键词项 w_{ik}，w_{jl} 的 TF-IDF 值占各自关键词项向量 TF-IDF 值总和的百分比确定.

①w_{ik} 和 w_{jl} 均属于 TSWT 中某一叶结点的词项有序队列.

②若 w_{jl} 属于 TSWT 中某一叶结点的词项有序队列，w_{ik} 不属于，但 w_{ik} 和 w_{jl} 具有较高的相似度(超过某个阈值 μ). 则在 w_{jl} 所属的词项有序队列中根据 w_{ik} 与当前词项有序队列中其它词项的相似度确定 w_{ik} 在词项有序队列中的次序. 反之亦然.

③w_{ik} 和 w_{jl} 均不属于 TSWT 中某一叶结点的词项有序队列，但 w_{ik} 和 w_{jl} 与 TSWT 中某一叶结点的词项有序队列中具有最大相似度以及最小相似度的关键词项的相似度均超过了某个阈值 μ，则在当前词项有序队列中根据 w_{ik} 和 w_{jl} 与其它词项的相似度确定 w_{ik} 和 w_{jl} 在词项队列中的次序.

④w_{ik} 和 w_{jl} 均不属于 TSWT 中某一叶结点的词项有序队列，但 w_{ik} 和 w_{jl} 具有较高的相似度(超过某个阈值 μ)并且 w_{ik} 和 w_{jl} 与 TSWT 中某一叶结点的词项有序队列中具有最大相似度以及最小相似度的关键词项的相似度都低于某个阈值 μ，则新建一个分支，将 w_{ik} 和 w_{jl} 插入该分支叶节点的关键词项队列.

(3) 文本相似度的计算.

根据式(2)并利用词项相似度加权树计算两个关键词项向量的相似度.

算法 1.　TSemSim.

输入:关键词项向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$,词项相似度加权树 TSWT,词项相似度阈值 μ

输出:关键词项向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 的相似度 $Sim(\boldsymbol{v}_i,\boldsymbol{v}_j)$

1. 初始化 TSWT.

2. 从向量 $\boldsymbol{v}_i$ 中的词项 w_{i1} 开始,寻找向量 $\boldsymbol{v}_j$ 中与 w_{i1} 最为相似的词项 w_{jk},记录词项 w_{i1} 和 w_{jk} 之间的相似度.根据 TSWT 的加权原则计算关键词向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 的加权因子 wf,同时依据 TSWT 的更新原则判断是否需要将词项 w_{i1} 和 w_{jk} 添加到 TWST.

3. 然后从向量 $\boldsymbol{v}_i$ 中的词项 w_{i2} 开始,重复步 2 的过程,直至向量 $\boldsymbol{v}_i$ 中所有的词项都在向量 $\boldsymbol{v}_j$ 中找到各自最为相似的词项,同时记录其相似度.

4. 累加步 2 和步 3 得到的相似度,除以向量 $\boldsymbol{v}_i$ 中词项的数量,即向量 $\boldsymbol{v}_i$ 的维度.以此作为向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 的相似度 $Sim(\boldsymbol{v}_i,\boldsymbol{v}_j)$.

5. 前面 3 个步骤是从向量 $\boldsymbol{v}_i$ 出发计算向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 之间的相似度,计算过程中找到了向量 $\boldsymbol{v}_i$ 中所有的词项在向量 $\boldsymbol{v}_j$ 中最为相似的词项,但向量 $\boldsymbol{v}_j$ 中的单词并没有找到向量 $\boldsymbol{v}_i$ 中与之对应的最为相似的单词.此外由于向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 的维度不一致,必须消除此影响.因此,还必须从向量 $\boldsymbol{v}_j$ 出发,重复步 2～4 的过程,得到向量 $\boldsymbol{v}_j$ 和 $\boldsymbol{v}_i$ 的相似度 $Sim(\boldsymbol{v}_j,\boldsymbol{v}_i)$.

6. 计算 $Sim(\boldsymbol{v}_i,\boldsymbol{v}_j)$ 和 $Sim(\boldsymbol{v}_j,\boldsymbol{v}_i)$ 的算术平均值,作为向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 的相似度.

7. 根据前述步骤记录的累加加权因子 wf 对关键词向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 的相似度进行加权处理后返回文本相似度.

TSemSim 算法中度量词项之间相似度的方法有许多文献对其进行了研究.大体上这些方法可以分为两类:基于语义网络或者知识库的方法以及基于从大量文本信息中学习到的信息模型方法.本文采用了 WordNet::Similarity[18] 工具包,该工具包实现了上述两大类共 8 种主流的词与词之间相似度计算的方法.文献[19]指出,基于信息内容度量的相似度方法优于其它方法.因此,本文采用了文献[14]所实现的相似度方法作为 TSemSim 算法中计算词项之间相似度的方法.

算法 TSemSim 中最主要的计算工作是计算两个词项间的相似度.如上所述,代表两篇文本的关键词向量 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 中各自的关键词数目分别为 m, n. 每两个词项 w_{jl},w_{ik} 之间都需要计算 $Sim(w_{jl},w_{ik})$ 来得到词项 w_{jl},w_{ik} 之间的相似度.因此计算两篇文本之间的相似度需计算 $m\times n$ 次 $Sim(w_{jl},w_{ik})$ 运算.对于一个文本数目较大的文本集合而言,这需要消耗大量运算时间.为减少算法的运行时间,我们事先将所有关键词向量中出现的关键词项以及关键词项两两之间的相似度 $Sim(w_{jl},w_{ik})$ 组织为一个 Hash 表,这样就可以将 $Sim(w_{jl},w_{ik})$ 运算转换为查找 Hash 表,从而有效的减少运算时间,提高算法的效率.

4　实　验

实验数据采用了业界广泛应用的 Reuters-21578 文本集合①以及 BBC 数据集②,这些数据集在文本的大小、聚类数目和文本分布都有着显著的差异.实验中分别选取了两个文本集合中各 3 个文本子集用于实验的验证,即来自于 Reuters-21578 的 Re1、Re2、Re3 和来自于 BBC 数据集的 BBC1、BBC2、BBC3.数据集合中的每一篇文本都预先被划分为一个或多个特定的类别.表 1 实验数据摘要总结了各个数据子集的特点.

表 1　实验数据摘要

数据集名称	聚类数目	总的文本数目	聚类中最少文本数目	聚类中最多文本数目	平均聚类文本数目
Re1	8	110	10	17	14
Re2	8	203	21	28	25
Re3	8	318	33	51	40
BBC1	5	200	35	50	40
BBC2	5	300	50	70	60
BBC3	5	400	80	80	80

实验首先采用了自然语言处理工具 LingPipe③对文本集合进行预处理,识别文本集合中的人名、地名、组织机构,同时给出语义标注.之后应用 TF-IDF 算法对文本中的名词进行 TF-IDF 的计算,从中选取特定比例的 TOP 关键词项,再结合本文提出的 TSemSim 文本相似度计算方法对文本进行相似度的计算,得到文本间的相似度矩阵.最后用得到的相似度矩阵与利用原始的 TF-IDF 算法以及基于文献[4]提出的算法(本文称之为 WordSim 算法)计算的文本相似度矩阵,进行聚类效果的比较.WordSim 算法最好的实验结果所采用的参数设置为:5 级上位关系的词项扩展、利用同义词项频率(即 Synset)代替词项频率以及词项的直接上下位关系词项进行词义消歧.后续的对比实验中设定上述参数进行相关实验.

实验中为了更客观地反映本文提出的文本相似度算法的有效性,聚类算法的实现采用了 CLUTO 工具包④.实验对比了 CLUTO 工具包实现的直接

① Reuters-21578 text categorization test collection, Distribution 1.0. Reuters. 1997. http://www.daviddlewis.com/resources/testcollections/reuters21578/

② BBC Dataset, Machine Learning Group. http://mlg.ucd.ie

③ LingPipe, Alias-i, Inc. http://www.alias-i.com

④ Karypis G. CLUTO—A clustering toolkit. Department of Computer Science, University of Minnesota. http://www.cs.umn.edu/~karypis/cluto/

K 均值(DKM)、二分 K 均值(BKM)以及凝聚 K 均值(AKM)等 3 种聚类算法.

实验采用了 F-度量值来衡量本文所提出的文本相似度.F-度量值是信息检索中一种组合查准率和召回率指标的平衡指标.实验计算出来的聚类结果使得我们能够检验每一篇文本在聚类后是否被划分为正确的类别以及同一个类别中是否包含了特定类别的文本.因此,可以计算每一个聚类 j 所属类别 i 的查准率 $P(i,j)$以及每一个聚类 j 所属类别 i 的查全率 $R(i,j)$.

设 n_i是类别 i 的文本数目, n_j是聚类 j 的文本数目,n_{ij}是聚类 j 中隶属于类别 i 的文本数目,则查准率 $P(i,j)$和查全率 $R(i,j)$可分别定义为

$$P(i,j)=\frac{n_{ij}}{n_j},\ R(i,j)=\frac{n_{ij}}{n_i}.$$

对应的 F 度量值 $F(i,j)$定义为

$$F(i,j)=\frac{2\times P(i,j)\times R(i,j)}{P(i,j)+R(i,j)}.$$

全局聚类的 F 度量值定义为

$$F=\sum_i \frac{n_i}{n}\max_j(F(i,j)).$$

其中 n 是文本集合中总的文本数目.通常,F 度量值越大,聚类效果越好.

实验中首先要确定 TSemSim 算法中选取不同比例的 TOP 关键词项对文本相似度计算的影响,实验设定关键词项相似度阈值参数 $\mu=0$,即不采用词项相似度加权树 TSWT 对关键词项加权,所有的关键词项都同等重要.此外,为了客观体现本文算法的真实性能,选取了 3 种 K 均值算法中最直接的 DKM 算法进行聚类.图 2 给出了在利用 DKM 聚类算法进行聚类的条件下,选取不同比例的 TOP 关键词项对相似度影响的实验结果.实验表明,如果选取文本中 30% 的 TOP 关键词项,能够取得最好的聚类效果,低于这个比例,由于选取的关键词项数目较少,导致提取的文本特征信息而使得聚类效果欠佳;超过这个比例,则由于选择的关键词项数目过多,不相关的关键词项之间较低的相似度拉低了文本之间的相似度,使得聚类效果随着关键词项数目增加而变得不理想.

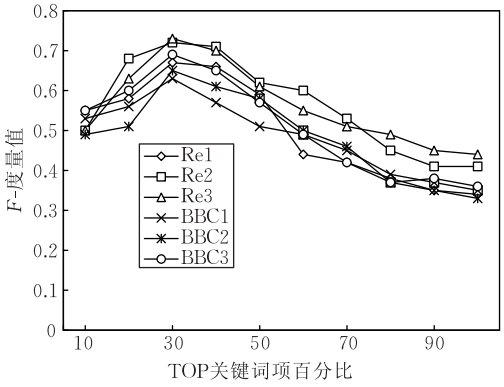


图 2 TOP 关键词项百分比对聚类的影响

接下来要确定 TSemSim 算法中关键词项相似度阈值参数 μ 对文本相似度计算的影响,图 3 给出了在选取了 30% 的关键词项作为文本特征向量,利用 DKM 聚类算法进行聚类的条件下,同一聚类中的关键词项相似度阈值 μ 的不同对聚类效果的影响.随着关键词项之间相似度阈值 μ 的逐渐升高,聚类效果也逐步提升.这是因为随着相似度阈值的提高,文本之间的区分度越来越大,使得聚类效果越来越好.但当相似度阈值在 0.7～0.75 之间到达聚类的最好效果时,再提高关键词项相似度阈值 μ 反而影响了聚类效果.F-度量值在词项相似度阈值超过 0.75 后迅速下降,这是因为在本文选取的关键词项相似度算法[14]对于两个关键词项的相似度计算很少能够有超过 0.75 的相似度值,导致词项相似度加权树的加权指导效果下降,降低了文本相似度计算的结果,因而在整体上降低了 F-度量值.

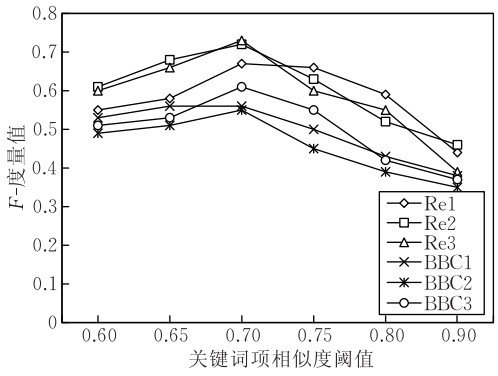
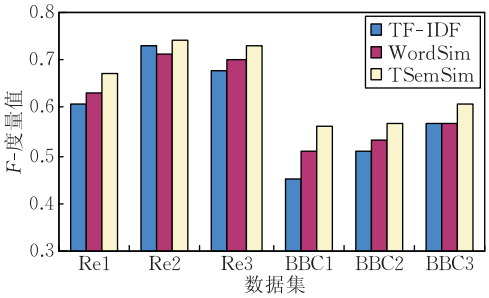
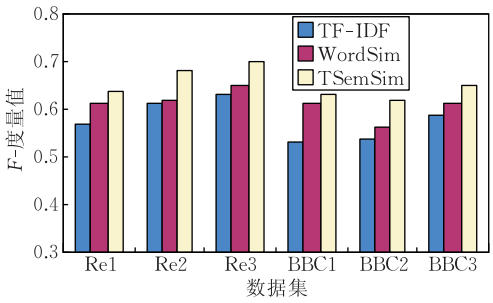


图 3 关键词项相似度阈值 μ 对聚类的影响

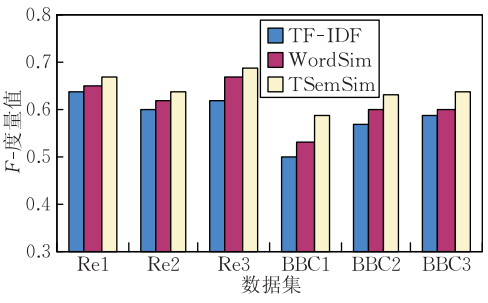
在设定词项相似度阈值 $\mu=0.75$,TOP 关键词项百分比为 30% 的情况下,采用 TSemSim 算法与 TF-IDF 以及 WordSim 算法的对比结果如图 4 所示.从图中可以得知,采用 TSemSim 算法计算的文本相似度在 3 种经典的聚类算法下都比采用传统的 TF-IDF 算法以及 WordSim 算法计算的文本相似度具有更好的 F-度量值.这表明在对文本相似度进行关键词项的选择以及语义相似度的计算上能够有效地改进聚类的效果.



(a) DKM 算法



(b) BKM算法



(c) AKM算法

图 4　TF-IDF、WordSim 和 TSemSim 计算的文本相似度在 DKM、BKM、AKM 聚类算法的 F-度量值对比

5　结　论

本文提出了一种新颖的结合文本词项语义的文本相似度量算法. 与传统的基于 TF-IDF 算法进行文本相似度量的计算方法不同,通过选取文本中具有较高 TF-IDF 值的关键词项,本文的文本相似度量方法有效减少了传统 TF-IDF 算法的计算量,同时有效地降低了传统向量模型表示文本所带来的高维影响. 在进行文本相似度的计算时,充分考虑了文本中关键词项在语义上的相似性,提出了一个用于指导文本相似度计算的词项相似度加权树的数据结构. 实验结果表明这种方法是有效的.

本文后续的研究将在现有探讨词项相似性的基础上,进一步深入分析文本相似度所蕴含的语义相似特征,考察文本句子、篇章等语义结构信息,更好地提高文本语义相似度的效果. 这对于即将到来的语义网的应用有着积极的意义.

参　考　文　献

[1] Fung B C M, Wang K, Ester M. Hierarchical document clustering//Wang John ed. The Encyclopedia of Data Warehousing and Mining, Idea Group. 2005: 970-975

[2] Salton G. The SMART Retrieval System-Experiments in Automatic Document Processing. Englewood Cliffs, New Jersey: Prentice Hall Inc, 1971

[3] Wang Y, Julia H. Document clustering with semantic analysis//Proceedings of the 39th Hawaii International Conferences on System Sciences. Hawaii, US, 2006: 54-63

[4] Hotho A, Staab S, Stumme G. Wordnet improves text document clustering//Proceedings of the Semantic Web Workshop at SIGIR-2003, 26th Annual International ACM SIGIR Conference. Toronto, Canada, 2003: 541-550

[5] Hall P, Dowling G. Approximate string matching. Computing Survey, 1980, 12(4): 381-402

[6] Coelho T, Calado P, Souza L, Ribeiro-Neto B, Muntz R. Image retrieval using multiple evidence ranking. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(4): 408-417

[7] Ko Y, Park J, Seo J. Improving text categorization using the importance of sentences. Information Processing and Management, 2004, 40(1): 65-79

[8] Erkan G, Radev D. Lexrank: Graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research, 2004, 22(7): 457-479

[9] Theobald M, Siddharth J, Paepcke A. SpotSigs: Robust and efficient near duplicate detection in large Web collections//Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Singapore, 2008: 563-570

[10] Han J, Kamber M. Data Mining: Concept and Techniques. 2nd Edition. San Francisco, CA, USA: Elsevier Inc, 2006

[11] Sussna M. Word sense disambiguation for free-text indexing using a massive semantic network//Proceedings of the 2nd International Conference on Information and Knowledge Management (CIKM'93). Washington, DC, US, 1993: 67-74

[12] Jiang J, Conrath D. Semantic similarity based on corpus statistics and lexical taxonomy//Proceedings of the International Conference on Research in Computational Linguistics. Taiwan, China, 1997: 19-33

[13] Ramage D, Rafferty A N, Manning C D. Random walks for text semantic similarity//Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing. Suntec, Singapore, 2009: 23-31

[14] Lin D. An information-theoretic definition of similarity//Proceedings of the 15th International Conference on Machine Learning. Madison, WI, US, 1998: 296-304

[15] Ramiz M A. A new sentence similarity measure and sentence based extractive technique for automatic text summarization. Expert Systems with Applications, 2009, 36(4): 7764-7772

[16] Gang L, Cheng Z, Li Z. Text information retrieval based on concept semantic similarity//Proceedings of the 5th International Conference on Semantics, Knowledge and Grid. Zhuhai, China, 2009: 356-360

[17] Tan P, Michael S, Vipin K. Introduction to Data Mining. Addison Wesley, US, 2005

[18] Pedersen T, Patwardhan S, Michelizzi J. Wordnet::Similarity-measuring the relatedness of concepts//Proceedings of the AAAI-04. San Jose, California, US, 2004: 38-41

[19] Budanitsky A, Hirst G. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures//Proceedings of the 2nd Meeting of the North American Chapter of the Assoc for Computational Linguistics. Pittsburgh, PA, USA, 2001: 29-34

HUANG Cheng-Hui, born in 1976, Ph. D. candidate, lecturer. His research interests include text mining, information retrieval, and natural language processing.

YIN Jian, born in 1968, Ph. D., professor, Ph. D. supervisor. His main research interests include information retrieval, machine learning, and data mining.

HOU Fang, born in 1975, Ph. D., lecturer. His research interests include computer architecture and storage system

Background

This work is supported by the National Natural Science Foundation of China (61033010), Research Foundation of National Science and Technology Plan Project (2008ZX10005-013), Research Foundation of Science and Technology Plan Project in Guangdong Province (2009A080207005, 2009B090300450, 2010A040303004)

How to build a document similarity model is critical to text mining. For our task, given two input text document, we want to determine a semantic similarity between them, thus our method goes beyond the simple word frequency based methods. Traditional word frequency methods model documents as TF-IDF vectors and use cosine similarity or Jaccard coefficient to compute similarity between documents. The TF-IDF vector ignores the meaning of words and the structure of documents. With TF-IDF model, users must process a vector set, which has large numbers of vectors and each vector has a dimension equals to words number, therefore inevitably leads to inefficient computing.

This paper improves on the state-of-the-art by combining TF-IDF with term semantic information in an integrated way: to analyze term significance and select those terms that have high TF-IDF values, then compute semantic similarity of these terms with external dictionary WordNet and Term Similarity Weight Tree. This method selects those terms with high TF-IDF value, therefore it can reduce dimension of document model effectively than traditional document model. At the same time, it analyzes semantic information of these terms and is closer to human's way to understand documents.

Our group has been working on the research of text similarity in text mining, and using several optimization technologies to compute the text similarity such as semantic information of words, word sequences and syntax structural information of document. Several papers have been published in respectable national journals.