基于最大关联规则的文本分类*)

何 玉 冯剑琳 王元珍
(华中科技大学计算机学院 武汉 430074)

摘 要 我们提出了一种新颖的、基于最大关联的文本分类方法—SAT-MOD+。在文本分类中,以往的方法在挖掘频繁项集和关联规则的时候,往往是将整个文本看作一个事务来处理的,然而文本的基本的语义单元实际上是句子。那些同时出现在一个句子里的一组单词比仅仅是同时出现在同一篇文档中的一组单词有更强的语义上的联系。基于以上的考虑,SAT-MOD+把一篇文档里的某些句子作为一个单独的事务。通过在标准的文本集上的大量实验,证明了 SAT-MOD+的有效性。

关键词 文本分类,关联规则,最大频繁项目集

Text Classification Based on Maximal Association Rule

HE Yu FENG Jian-Lin WANG Yuan-Zhen
(Department of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract We propose a novel association based method called SAT-MOD+ for text classification. While previous methods mainly mined frequently co-occurring words (frequent itemsets) at the document-level, the basic semantic unit in a document is a sentence. Words within the same sentence are typically more semantically related than words that appear in the same document. Our proposed SAT-MOD+ views a sentence rather than a document as a transaction. The effectiveness of proposed SAT-MOD+ method has been demonstrated by extensive experimental studies using popular benchmark text collections.

Keywords Text classification, Association rules, Maximal frequent itemsets

1 引言

随着电子化文本的持续增长,文本分类(Text Classification 或 Text Categorization)这个经典的问题在我们的日常生活中变得越来越重要。文本分类技术被广泛应用于文本检索、文本组织、邮件过滤、Web 网页分层等领域。各种文本分类方法已经陆续被提出来,如贝叶斯网络(Bayesian Networks)、决策树(Decision Trees)、神经网络(Neural Networks)、支持向量机(Support Vector Machines)以及基于关联规则的分类方法[3~5]等。

传统的基于关联规则的分类算法都是基于文档级的,文档中的单词被看作是项目(item),每一篇文档被看作是一个事务(transaction),即项目的集合。在不同文档中频繁出现的项目集(即:文档级的单词共同出现)被用于捕捉文本的语义和产生分类的规则。然而,文档中的基本语义单元其实是句子。在同一个句子中共现的单词通常以这样或那样的方式相互关联,与跨越文档中多个句子的同一组单词相比,它通常更有意义。前面我们提出了基于句子级别的关联规则对文本进行分类的方法,SAT-Mod[1,2],它将一个自然语句看作是一个关联事务,而一篇文档则看作是它所有自然句子的集合。本文介绍了新提出的 SAT-MOD+算法,SAT-MOD+使用了最大关联规则来表示文档,从而减少了规则的个数,提高了分类的速度,并且通过在 Reuters 上的实验,验证了 SAT-MOD+的有效性。

2 SAT-MOD+

用于文本分类的关联规则具有如下形式:"c ==> Category",例如:money AND advertisements ==> spam。据我们所知,目前所有的基于关联规则的邮件分类方法基于如下思想:邮件中的单词被看作是项目(item),例如:"money"和"advertisement"是两个不同的项目;每一篇文档被看作是一个事务(transaction),即项目的集合。在不同文档中频繁出现的项目集被用于捕捉文本的语义和产生分类的规则。

然而,文档中的基本语义单元其实是句子。在同一个句子中共现的单词通常以这样或那样的方式相互关联,与跨越文档中多个句子的同一组单词相比,它通常更有意义。基于上述观察,我们将一个句子而不是一篇文档作为基本的语义单元。

2.1 基本定义

在日常生活中,人们通常倾向于通过在不同的句子中重复某些词来加强某些中心思想,因此那些经常出现的词表达了整个"文档主题"的某一方面。为了运用频繁项目挖掘算法来发现这些频繁出现的单词,我们给出如下的基本定义。

令 $W=\{w_1,w_2,\cdots,w_M\}$ 为单词的集合,称为项目。W 中一组项目的集合称为项目集,包括仅仅只有单个项目的项目集的情况。我们把项目集中单词的数目称为项目集的长度,称长度为 k 的项目集为 k-项目集。一个事务被定义为以".","!","?",和";"等句子结束符结束的一条自然语句。

*)基金项目:国家自然科学基金(编号:60373000)。何 玉 硕士研究生,主要研究方向:文本分类;冯剑琳 博士,主要研究方向:数据库,联机事物处理、数据挖掘;王元珍 教授,博士生导师,研究方向:数据库和多媒体技术的理论研究及实现技术。

如果项目集 I 是事务 T 的子集，我们说 T 包含 I。如果文档 D 中 S_d 个事务包含 I，那么项目集 I 在文档 D 中的文档支持度(document Support)为 S_d。

定义 1(文档频繁项目集，DFI) 项目集被认为是文档 D 的 DFI(文档频繁项目集)，如果其文档支持度大于 2(即文档最小支持度)。

注意到 Apriori 算法中使用的频繁项目集的属性：任何频繁项目集的子集也是频繁项目集。因此，最大长度的 DFIs 可以表示所有的 DFIs。我们定义最大 DFI 如下：

定义 2(最大文档频繁项目集，MDFI) 如果一个 DFI 不是当前文档的任何 DFI 的子集，则我们说该 DFI 是最大的。

我们可以从一篇文档的某些句子中挖掘出所用的 MDFIs，然后利用这些 MDFI 来表示文档。我们采用基于 FP-Tree 的 FPMax[8] 算法来挖掘 MDFI。

2.2 类频繁项目集

我们用 MDFIs 来产生类频繁项目集以捕捉同一类下不同文档之间的"公有文档主题"，"公有文档主题"代表了整个"类主题"的某个方面。

如果项目集 I 在类 C 的 s_c 个文档中是文档频繁的，项目集 I 在类 C 中的支持度为 s_c。当项目集 I 在文档 D 中是文档频繁的，称 D 为 I 所覆盖，同时 D 也被称为 I 的支持文档(supporting document)。

定义 3(类频繁项目集，CFI) 类 C 的类频繁项目集(CFI)是指类 C 的项目集，它在类 C 中的类支持度大于用户指定的最小支持度(称为 category minsup，类最小支持度)。

自然地，一个类的最小支持度应该设为 2 以保证一个 CFI 至少在类的 2 篇文档中是频繁的。然而如果没有充足的训练文档，单独一篇文档也可能代表了"类主题"的一个方面。因此，我们将类最小支持度的默认值设为 1。

给定 M 个预定义类 $\{C_1, C_2, \cdots, C_m\}$ 的集合，我们假定每一个类 C_i 包含 K_i 个训练文档。类 C_i ($1 \leqslant i \leqslant M$) 的关联规则是形为 $I \Rightarrow C_i$ 的蕴涵式，其中 I 是 C_i 的 CFI。I 可以是几个类的 CFI，也可以是某个类的一些文档的 DFI 却不是该类的 CFI，因此，我们需要确定，对整个训练文档集而言，其预计分类的置信度。为了定义 CFI 预计分类的置信度，我们提出了一种新颖的自启发式算法 moderate itemset fittest (缩写为 MODFIT)[2] 用于获取具有最高置信度的项目集。

定义 4(类频繁项目集在类 C_i 中的置信度) 类频繁项目集 I 在类 C_i 中的置信度，表示为 $\mathrm{Conf}(I \Rightarrow C_i)$，定义为 S_i 与 S_{tot} 的比值，其中 S_i 为 I 在 C_i 中的支持度，S_{tot} 为整个训练文档集中，I 所覆盖的不同的支持文档的总数，即 $\mathrm{Conf}(I \Rightarrow C_i) = S_i / S_{tot}$。

对于短的项目集而言，S_i 和 S_{tot} 可能同时高；而对于长的项目集，S_i 和 S_{tot} 可能同时低。因此，我们希望通过"适中的项目集"获得 S_i 和 S_{tot} 最好的折中。

2.3 类前缀树

给定类 C 的 K 个训练文档和用户指定的类最小支持度 c_s，我们现在说明如何使用类前缀树来收集类(频繁)项目集。在类前缀树中，每一条边标记为一个项目，每一个节点包含了从根节点到该节点的路径上的边所对应的项目集合。简单起见，我们将在下文中交替使用节点和(它对应的)项目集。例如，图 1 表示了一棵类前缀树。

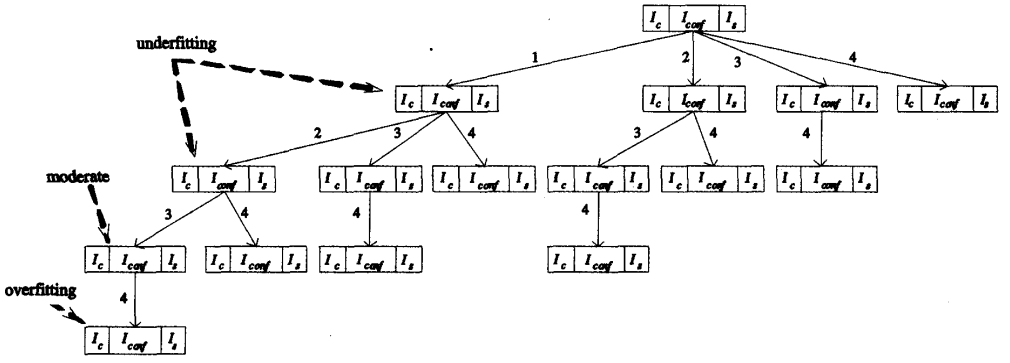

图 1 A prefix-tree for the four items 1, 2, 3, and 4

如图 1 所示，每一个节点包含 3 个计数器：I_c，I_{conf} 和 I_s。我们使用 I_c 和 I_{conf} 分别表示节点所对应的项目集 I 的类支持度和置信度。计数器 I_s 用于分类阶段，统计项目集 I 在新文档 D_u 中的文档支持度。

给定 M 个预定义的类，我们逐个构造 M 棵类前缀树。同时，我们构造一棵全局前缀树以收集在所有训练文档集中发现的每一个类项目集。这棵全局树用于计算 S_{tot}，它表示在整个训练文档集中类项目集覆盖的不同支持文档的总数。

一旦全部 M 棵类前缀树及全局树构造完毕，我们使用定义 3 计算每一棵类前缀树节点的置信度(I_{conf})，此后，置信度为百分比的形式。

然后我们使用 MODFIT 算法来修剪每棵前缀树。直觉上，MODFIT 修剪等价于沿着类前缀树的根结点开始的路径扩展空的项目集每次一个项目，直到项目集的置信度不再有任何增长。

获得全局适中项目集后，我们得到了每一个叶子节点都保存了一个适中的 CFI 的类前缀树，其每一个内在节点都被标为低配。

2.4 得分模型

给定一个预定义的类集 $\{C_1, C_2, \cdots, C_M\}$ 和它们对应的类前缀树集 $\{CPT_1, CPT_2, \cdots, CPT_M\}$，定义每棵 CPT_i ($1 \leqslant i \leqslant M$) 和待分类文档 D_u 之间共享的"公共项目集"为类 C_i 和 D_u 的类交集(记为 $CPT_i \cap D_u$)。

我们将 CPT_i 视作一个线性分类器，并依此计算 D_u 与类 C_i 之间的相似度：我们将 D_u 视为一个矢量，每一个公共项目作为 D_u 的一维，将 I_{conf} 作为每一维的权重。为使 D_u 与不同

类之间的相似度具有可比性，我们需要解决以下问题：通常 D_u 在不同的类上有不同的类交集，不同的公共项目集具有不同的文档支持度。我们分两步解决上述问题。首先我们用如下公式规范化不同的文档支持度：

$$s(C_i|D_u)=\frac{\sum I_{comf}\times I_s}{\sum I_s}\times[I\in CPT_i\cap D_u]$$

其中$[I\in CPT_i\cap D_u]$为 Iverson 常数，即如果项目集 I 属于 $CPT_i\cap D_u$，它取值 1，反之取值 0。

然后，我们用规范化的类交集大小乘以 $s(C_i|D_u)$来得到平均得分。最终的得分函数如下：

$$sim(C_i|D_u)=s(C_i|D_u)\times\frac{|CPT_i\cap D_u|}{\max_{j\in\{1,\cdots,M\}}|CPT_j\cap D_u|}$$

3 实验研究

3.1 文本数据集与预处理

Reuters-21578[7]数据集：ModApte 划分，它包括 9603 篇训练文档和 3299 篇测试文档。自然地，如果一个类的训练文档太少，我们不能期望分类有很好的效果。因此，许多研究者主要选择了 Reuters 语料中最大的 10 个分类进行实验研究。我们也采用了这个策略，将该训练集用于多分类任务。

对语料所做的数据预处理如下：将每一篇文档中的单词全部换成小写，所有的由数字字符构成的单词和长度超过 24 个字符的过长的单词被过滤掉。句子切分时，我们采用“.”、“?”、“!”和“;”作为句子切分的分隔符；段落切分时，我们采用“.”加回车符作为段落切分的分隔符。同样，超过 30 个词的过长句子，被人为地切分成每句 30 个单词的多个句子以处理没有结束符的表格的情况。对每一个训练文档，我们进行了去停词[6]、提取词干[6]、编码、分句等预处理工作。

3.2 度量标准

我们使用 micro-averaged and macro-averaged BEP 来衡量分类的精度。在多分类的情况下，我们首先计算文档 D_u 和所有类别的相似度，将相似度最高的类标签赋予 D_u，然后我们通过设置一个百分比参数 labelMinsup，如果 D_u 和某个类别的相似度超过最高相似度与 labelMinsup 的乘积，我们就将该类的泪标签赋予 D_u。

3.3 实验结果及分析

SAT-MOD+有 2 个参数：catMinsup(类最小支持度)，labelMinsup(分类百分比)，我们设定 catMinsup 的值为 1～5，labelMinsup 的值为 50%，55%，…，90%，采用在训练及上进行 10-cross validation 的方式确定出最好的参数组合：catMinsup 的取值为 1，labelMinsup 的取值为 55%和 60%。表 1 列出了在此参数组合下的分类结果与其他分类算法的精度比较的结果。

ARC-BC 是目前已知的最好的文档级的关联文本分类算法。从表 1 可以看出 SAT-MOD+的分类效果优于 ARC-AB 和其他除了线形 SVM 的分类算法。在类“corn”和“wheat”中效果较差，而在类“grain”中的效果较好。其原因可能是这些类太相似而难于区分。事实上，类“corn”和类“wheat”是类“grain”的子类，许多文档仅仅被正确分到父类，而没有被分到任何子类。类似的情况在类“interest”和“money-fx”之间存在。

在分类效率上，由于使用了最大关联规则，经过修剪后的规则数比使用一般关联规则大大减少，因而训练及测试时间都有显著的缩短。SAT-MOD+在约 54 秒的时间内就可以完成所有的训练和测试，需要的内存约 50M。

表 1 Reuters 中最大的 10 个类的分类 BEP 值

BEP	SAT-MOD		ARC-BC		Bayes	Rocchio	C4.5	k-NN	LinearSVM
	lableMinsup		δ=50						
	55%	60%	10%	15%					
acq	90.3	90.7	90.9	89.9	91.5	92.1	85.3	92.0	93.6
com	71.9	68.5	69.6	82.3	47.3	62.2	87.7	77.9	90.3
crude	87.3	88.3	77.9	77.0	81.0	81.5	75.5	85.7	88.9
carn	92.4	93.3	92.8	89.2	95.9	96.1	96.1	97.3	98.0
grain	87.7	87.4	68.8	72.1	72.5	79.5	89.1	82.2	94.6
interest	76.3	75.6	70.5	70.1	58.0	72.5	49.1	74.0	77.7
mony-fx	78.7	79.5	70.5	72.4	62.9	67.6	69.4	78.2	74.5
ship	86.5	85.6	73.6	73.2	78.7	83.1	80.9	79.2	85.6
trade	78.1	79.5	68.0	69.7	50.0	77.4	59.2	77.4	75.9
wheat	74.5	76.4	84.8	86.5	60.6	79.4	85.5	76.6	91.8
micro-avg	87.6	88.2	82.1	81.8	72.0	79.9	79.4	82.3	92.0
macro-avg	82.4	82.5	76.74	78.24	65.21	79.14	77.78	82.05	87.10

总结 在本文中，我们提出 SAT-MOD 的改进算法 SAT-MOD+。SAT-MOD+试图使用在文档的同一句子中同时出现的最大频繁项目集来表示句子，进而表示整个文档。SAT-MOD+的效果已被证明同那些著名的分类算法是可比的，并且远远好于以前的基于关联的分类方法。我们下一步的工作是将 SAT-MOD+方法运用到更为复杂的数据集比如 20NG 数据集上。另一个可能的方向是将 SAT-MOD+的基本思想运用到其他以前将文档视为一个事务的分类方法中，比如 Large Bayes 或 FIHC。

参 考 文 献

1 Feng Jianlin, He Yu, Zou Jing. Moderately Extending Core Words for Text Classification. Submitted to SIGKDD06

2 Feng Jianlin, Liu Huijun, Zou Jing. SAT-MOD: Moderate Itemset Fittest for Text Classification. In: WWW. Invited Poster Paper, 2005. 1054～1055

3 Sebastiani F. Machine Learning in Automated Text Categorization. ACM Computing Surveys, 2002, 34(1): 1～47

4 Dumais S, Platt J, Heckerman D, Sahami M. Inductive Learning Algorithms and Representations for Text Categorization. In: CIKM98, 1998. 148～155

5 Liu B, Hsu W, Ma Y. Integrating classification and association rule mining. In: SIGKDD, 1998. 80～86

6 Frakes W B, Baeza-yates R. Information Retrieval: Data Structures and Algorithms. Prentice-Hall, 1992. ftp://ftp.vt.edu/pub/resue/IR.code/

7 http://www.daviddlewis.com/resources/testcollections/reuters21578/

8 Grahne G, Zhu J. Efficiently Using Prefix-trees in Mining Frequent Itemsets. In: Proc. FIMI, 2003