基于String Kernel和KPCA的负实例语法特征提取算法

吕 威[1,2],林文昶[1],姚正安[1],李 磊[1]
LV Wei[1,2],LIN Wen-chang[1],YAO Zheng-an[1],LI Lei[1]

1.中山大学 软件研究所,广州 510275
2.北京师范大学 珠海分校 信息技术学院,广东 珠海 519085
1.Software Research Institute,Zhongshan University,Guangzhou 510275,China
2.School of Information Technology,Beijing Normal University Zhuhai Campus,Zhuhai,Guangdong 519085,China
E-mail:luwei00@126.com

LV Wei,LIN Wen-chang,YAO Zheng-an,et al.False instance grammatical feature extraction algorithm based on String Kernel and KPCA.Computer Engineering and Applications,2009,45(20):136-139.

Abstract: This paper presents a method that translates false instance in grammatical database to kernel matrix through String Kernel,and uses KPCA to extract feature of the translated kernel matrix.We can separate the original false instance database into many small characteristic tables according to these extracted features,and design a classified machine by constructing false instance characteristic table.A new sentence is distributed to some characteristic table for matching of false instance through this classification machine.For characteristic table is much little than original false instance database,the running speed is enhanced very much without decreasing the accuracy of grammatical check.By compared with grammar inspection function of word,the new system demonstrates more quick speed while keeping the accuracy of grammatical check.

Key words: String Kernel;Kernel Principal Component Analysis(KPCA);false instance;feature extraction

摘 要: 提出通过String Kernel方法把负实例语法数据库中的负实例转化成核矩阵，再用Kernel Principal Component Analysis(KPCA)对转换的核矩阵进行特征提取,进而可将原始负实例数据库按照这些特征分成多个容量较小的特征表。通过构造负实例特征索引表设计了一个分类器,待检查的句子通过此分类器被分配到某个负实例特征表里进行匹配搜索,而此特征表的特征属性数和记录数要远远小于原始负实例数据库中的相应数目,从而大大提高了检查的速度,同时不影响语法检查的精度。通过比较测试,可看出提出的方法在保证语法检查精确度的同时有更快的速度。

关键词: String Kernel;核主成分分析;负实例;特征提取
DOI:10.3778/j.issn.1002-8331(2009)20-0136-04 **文章编号:**1002-8331(2009)20-0136-04 **文献标识码:**A **中图分类号:**TP311

1 引言

文本中的语法检查是自然语言处理的主要应用领域之一[1-2],语法检查通常分成字词级的拼写检查、语法级错误检查、语义级错误检查和语境级的错误检查。目前,对字词级错误的检查已经有比较充分的研究[3-4],相比之下,对语法和语义层次错误检查的研究尚不够深入。

语法层次错误检查的方法大体可分为两类:基于规则的分析方法和基于统计模型的方法[5-6]。而这两种方法有一个共同的特点,都是以词性标注系统为基础的。词性标注系统对句子中单词的标记是以“所要标注的句子是正确的”为前提。但即使句子是正确的，现有的词性标注系统的标注准确率也不是100%的,一般的准确率为90%左右。这样就会影响到语法错误检查的准确率,更严重的会导致把一些原本正确的句子,由于词性标注的不准确而误判。例如对于错误“for the purpose to compute”,正确的应该是“in order to compute”,原来提出的一些方法并不能检查出来。

提出一个基于错误实例(负实例)[7-8]和错误特征相结合的方法,主要根据最相近负实例查错,不再以词性标注系统为基础。该方法抛开了词性标注系统所带来的不确定性,可大大减少错误判别的出现概率。但搜集的负实例表是非常庞大的,如果每检查一个待检的句子就要去查找匹配相应的整个负实例表，需要大量的时间。其实很多负实例都具有相同的特征,如“for the purpose to compute”这个负实例,负实例表中还有“for the purpose to copy”、“for the purpose to cook”等,这些实例都具有“for the purpose to”这样的特征,可以提取出来。

首先通过String Kernel[9-10]方法把负实例语法数据库中的负实例转化成核矩阵,再用Kernel Principal Component Analysis(KPCA)[11]对转换的核矩阵进行特征提取,进而可将原始负实例数据库按照这些特征分成多个容量较小的特征表。结合String Kernel和KPCA为语法检查系统设计了一个分类器,待

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.10171113,No.10471156)。
作者简介:吕威(1978-),男,博士研究生,讲师,主要研究领域为数据分析,机器学习。
收稿日期:2008-10-10 **修回日期:**2008-11-18

检查的句子通过此分类器被分配到某个负实例特征表里进行匹配搜索,而此特征表的特征属性数要远远小于原始负实例数据库中的特征数,从而大大提高了检查的速度,同时不影响语法检查的精度。通过比较测试,提出的方法在保证语法检查精确度的同时有更快的速度。

2　相关工作

2.1　基于负实例的语法检查方法

基于负实例的语法检查方法利用了基于案例推理(Case-Based Reasoning,CBR)[7]的方法,就是新来一个待检查的文档,到语法负实例数据库中去查找匹配,如果找到相近的负实例,则表明新来的文档中具有同样的错误。

基于案例推理 CBR 的核心思想是重用过去人们解决问题的经验解决新问题。推理循环一般由 4 个基本过程组成,即 4R 循环:Retrieve、Reuse、Revise、Retain,分别对应着案例的提取、重用、改编和保存(学习)。当给定一个待求解问题,CBR 首先检查是否存在一个同样的训练案例。如果找到一个,则返回附在该案例的解上。如果找不到同样的案例,则 CBR 将搜索具有类似于新案例的训练案例。概念上讲,这些训练案例可以视为新案例的邻接者。CBR 是对已经发生的"历史"与待求解问题进行相似性匹配,利用相似的一个或若干个"历史"对待求解问题进行解答。

CBR 有多种测量相似度的测度方法,最常用的是来自 Minkowski 方法测度,其公式为:

$$D(c_i,c_j)=\left(\sum_{k=1}^{n}\left|a_{ik}-a_{jk}\right|^{r}\right)^{\frac{1}{r}},r\geqslant 1$$

其中 c_i、c_j 为案例 i 和 j,a_{ik}、a_{jk} 为案例 i 和 j 的第 k 个属性,每个案例有 n 个属性。

2.2　Kernel 方法介绍

按照广义线性判别函数的思路,要解决一个非线性问题,可以设法将它通过非线性变换转化为另一个高维空间中的线性问题,在这个变换空间对问题进行求解。这里,把这个变换空间称为特征空间,把这种变换称为映射[11-12]。如果引入 Kernel 核函数

$$k(x,y)=\phi(x)\cdot\phi(y)$$

则可使得在这些高维的特征空间中只需进行点积运算而不必明确求出映射函数。

一般的核有:多项式核 $k(x,y)=(x\cdot y)^d$、高斯(RBF)核 $k(x,y)=\exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right)$以及 Sigmoid 核 $k(x,y)=\tanh(k(x\cdot y)+\theta)$ 等。String Kernel 是特别针对字符设计的核函数。

3　基于 String Kernel 和 KPCA 的负实例语法特征提取算法

3.1　基于 String Kernel 的核相关矩阵计算算法

String Kernel 方法可以用来对比两个字符串实例的相似度,即比较这两个实例之间含有多少个相同的子串,相同的子串越多,则它们就越相似。需要用延迟因子 λ 来辅助设定不同的权值,权值可以根据子串在字符串里面的跨度来设定,连续存在子串的权值大,反之就较小。例如对比"car","cat","bat"和"bar"的相似程度。假设子串长度 k=2;则有一个 8 维的特征空间 F。其中 $\lambda\in(0,1)$。

表 1　使用 String Kernel 计算字符串映射值例子

函数值	子串							
	c-a	c-t	a-t	b-a	b-t	c-r	a-r	b-r
$\phi(cat)$	λ^2	λ^3	λ^2	0	0	0	0	0
$\phi(car)$	λ^2	0	0	0	0	λ^3	λ^2	0
$\phi(bat)$	0	0	λ^2	λ^2	λ^3	0	0	0
$\phi(bar)$	0	0	0	λ^2	0	0	λ^2	λ^3

其中 $\phi:D\rightarrow F$ 是计算单词包含上述的这些 k=2 的子串的权值。$\phi(D)\subseteq F$ 是一个八维的数据,如 $\phi(cat)=(\lambda^2,\lambda^3,\lambda^2,0,0,0,0,0)$,其中 λ 值的指数取值 l 是子串在单词里面的跨度。例如单词"cat"里面,"ca"在"cat"里面是连续的,单词跨度为 2,则 $\phi(cat)$包含"ca"的权值为 λ^2;而"ct"在单词"cat"里面从 c 到 t 的跨度为 3,$\phi(cat)$包含"ct"的权值为 λ^3。这样把跨度作为 λ 的指数就可以反映单词的相关信息,连续的子串的映射函数值(权值)较大,而不连续的子串的信息也没有丢失。

计算完每个字符串的 ϕ 函数值后,可使用 ϕ 函数值的内积来定义核函数 K 为对比两个字符串相似度的函数。如"car"和"cat"的相似度可用 $K(car,cat)=\phi(car)\cdot\phi(cat)=\lambda^4$ 来衡量,即 K 值为两个单词的 ϕ 函数值内积和。

算法 1　核相关矩阵的计算算法。

输入:负实例表 f(设共有 n 条记录),延迟因子 λ,子串长度 k。

步骤 1　根据 k 值确定特征空间的维数 d。

步骤 2　根据某一子串在负实例中的跨度 l,计算负实例包含该子串的 ϕ 函数映射值 λ^l。

步骤 3　所有的 ϕ 函数映射值构成一个 $n\times d$ 的延迟矩阵 $\boldsymbol{T}$。

步骤 4　对 $\boldsymbol{T}$ 中任两个行向量 $\boldsymbol{T}_i$,$\boldsymbol{T}_j$,$1\leqslant i\leqslant n$,$1\leqslant j\leqslant n$,$i\neq j$,计算它们的核函数值:

$$K(\boldsymbol{T}_i,\boldsymbol{T}_j)=\boldsymbol{T}_i\cdot\boldsymbol{T}_j$$

输出:$n\times n$ 的核相关矩阵 $\boldsymbol{K}$。

3.2　基于 String Kernel 和 KPCA 的负实例语法特征提取算法

由于得出的核矩阵规模很大,为了避免维度灾难以及计算上的困难,必须要进行"降维"处理,而降维的标准就是要保留尽可能多的原始数据的相关信息。主成分分析(PCA)是其中应用非常广泛的一种方法,采取找出几个综合因子来代表原来众多的变量的方法,使得这些综合因子尽可能地反映原来变量的信息量。

传统的 PCA 是线性的,而 KPCA[11]非常适于提取数据中有趣的非线性结构。把数据 $x_1,x_2,\cdots,x_n\in R^N$ 映射到特征空间,计算协方差矩阵:

$$\boldsymbol{C}=\frac{1}{n}\sum_{j=1}^{n}\phi(x_j)\phi(x_j)^{\mathrm{T}}$$

然后通过解特征值问题计算出主分量:找到 $\gamma>0$,$\boldsymbol{V}\neq 0$,满足:

$$\gamma\boldsymbol{V}=\boldsymbol{C}\boldsymbol{V}=\frac{1}{n}\sum_{j=1}^{n}(\phi(x_j)\cdot\boldsymbol{V})\phi(x_j)$$

采用了 KPCA 方法来对从算法 1 计算出的核相关矩阵进行降维,则问题可转化成:

找到 $\gamma>0$,$\boldsymbol{V}\neq 0$,满足 $\gamma\boldsymbol{V}=\boldsymbol{K}\boldsymbol{V}$　(1)

其中 γ 为特征值，$\boldsymbol{V}$ 为特征向量，$\boldsymbol{K}$ 为算法 1 输出的核相关矩阵。通过式(1)，可得到一个规模小得多但又保留了大部分信息的主成分矩阵。通过 KPCA 分析后，每个类的特征提取也很容易实现，把特征提取出来后，就可以构造出特征索引表。具体操作如图 1 所示：

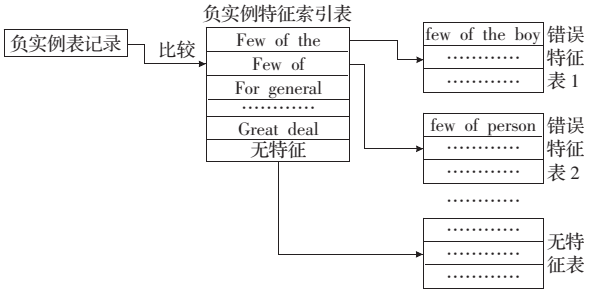


图 1 提取负实例特征表的过程示意图

总结以上步骤，可以得出基于 String Kernel 和 KPCA 的负实例语法特征提取算法。

算法 2 基于 String Kernel 和 KPCA 的负实例语法特征提取算法。

输入：负实例表 f(设共有 n 条记录)，延迟因子 λ，子串长度 k。

步骤 1 根据算法 1 计算核相关矩阵 $\boldsymbol{K}$。

步骤 2 根据公式(1)来计算核矩阵 $\boldsymbol{K}$ 的特征值并得出主成分矩阵。

步骤 3 根据主成分矩阵，提取各个类的特征。

步骤 4 按图 1 方法，根据这些特征对原始负实例表进行特征表的构造。

输出：多个特征表。

通过多个输出的特征表其实已经设计好了一个分类器，对新来待检查的文档，只要根据算法 1 计算出其 ϕ 函数映射值，则可以用基于负实例推理中的公式计算，从而知道其应该属于哪一个错误特征表，得出其有无错误。

4 实验结果

在大量数据上进行了算法实验，实验结果表明，提出的算法有较好的实用能力和效果。

4.1 负实例数据集

负实例表 f，共有 3 373 条负实例，对其用第 3 章提出的算法进行操作。子串的长度 k 取为 2(长度 3 以上的由于计算量太大，暂不考虑)，英文共有 26 个字母，则长度为 2 的子串有 676 个："aa"，"ab"，… "az"，"ba"，"bb"，…"zz"，即特征空间维数为 676，取 λ=0.7，根据算法 1 可以得出一个 3 373*676 的延迟矩阵 $\boldsymbol{T}$。因为要求记录要严格相似，所以对跨度大于 2 的子串 ϕ 映射函数值一概取为 0。

例如对于记录"focus can be made on"，有

维坐标	0	…	3	…	13	…	31	…	53	…	675
对应子串	aa	…	ad	…	an	…	be	…	ca	…	zz
ϕ 映射值	0	…	0.49	…	0.49	…	0.49	…	0.49	…	0

从 $\boldsymbol{T}$ 出发就可以计算各个负实例的核相关矩阵 $\boldsymbol{K}$ 的函数值，如

$\boldsymbol{K}$("focus can be made on","focus had to be made on")=

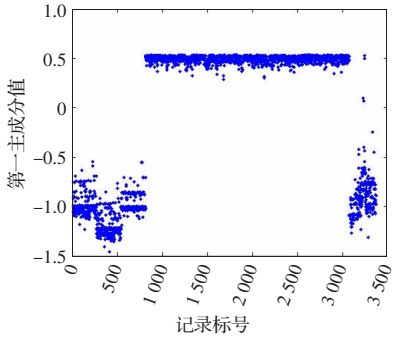
ϕ("focus can be made on")·ϕ("focus had to be made on")=

3.563 7

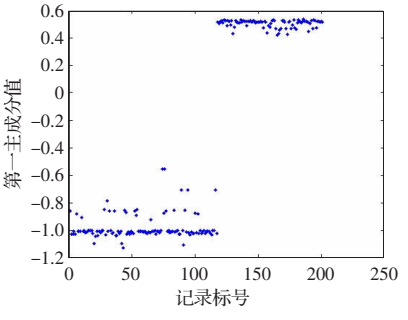
这个值就是两个错误子串的相似系数，由所有相似系数组成的 3 373*3 373 的矩阵就构成了负实例表 f 的核相关矩阵。

计算出核相关矩阵后，使用 KPCA 进行主成分的提取。通过分析可知前 50 个主成分方差占总体方差的 71.93%，前 100 个主成分占 85.43%，而第一主成分占 26.20%。

图 2 是错误记录(负实例)用 KPCA 提取出的第一主成分结果。



(a)3 773 条记录(整个表 f)



(b)记录号 700~900 的点(200 个点)

图 2 对负实例核矩阵用 KPCA 处理生成第一主成分结果
(横坐标为记录标号，纵坐标第一主成分值)

由图 2(a)可见大部分记录在第一主成分的值上是很相似的，而且主要分成了两块，在第 800 个记录左右，存在明显的断面。为了放大显示两类不同特征的记录的断面，图 2(b)缩小了记录的范围，取了 200 个点。

根据这些记录的编号，可查到第 816 条记录为"for general suppress"，而 700~816 记录之前的这些记录都是"for general…"这种结构；817 条记录为 "for the purpose to abate"，817~900 的这些记录基本都是"for the purpose to…"这种结构。可以看出，主成分分析对于记录的特征显示是明显而准确的。

通过 KPCA 分析后，每个类的特征提取就容易实现了，以下就是从负实例中提取出来的一些错误特征：

few of the　few of　for general　for the purpose to

great deal　focus to be made on …

把这些特征提取出来后，就可以构造出特征索引表。

4.2 实验比较

可以证明提出的算法不会降低检查精度，因此比较的重点在于运行速度。本文的实验重点关注文[8]中的 wordhelp 方法和本文提出算法的比较，而 wordhelp 与 word 的比较具体可见文[8]。分别比较在含有 0%、25%、50%错误语法的文章数据集

上的结果，文章长度不是固定不变的。本实验都在 Windows XP,1 G 内存,CPU P4 3 G 上运行。实验结果表明,本文提出的算法有更快的运行速度。

表 2　原方法和现方法在不同文章数据集上的运行速度

文章数据集	文章长度(单词数)	运行时间/s	
		Wordhelp	本文算法
0%	100	16.12	4.09
	200	39.58	8.06
	300	59.86	12.95
25%	100	12.53	5.03
	200	26.38	8.82
	300	40.31	14.60
50%	100	8.70	5.33
	200	20.30	8.39
	300	30.23	14.09

按照实验的结果可以看出,无论含错误的比例和长度是多少,本文方法都比 wordhelp 方法的运行时间要快得多,而且,随着文本长度的增加,这种优势会更加明显。wordhelp 方法随着错误的增多,错误的检查速度会有所提高,这是因为对于含有错误越多的文本,需要遍历整个负实例表的可能性越低,用时越少。

5　结束语

研究了基于负实例和错误特征相结合的语法特征提取方法,主要根据最相近负实例查错,不再以词性标注系统为基础,但搜集的负实例表是非常庞大的。首先通过 String Kernel 方法把负实例语法数据库中的负实例转化成核相关矩阵，再用 KPCA 对转换的核相关矩阵进行特征提取,进而可将原始负实例数据表按照这些特征分成多个容量较小的特征表。结合 String Kernel 和 KPCA 为语法检查系统设计了一个分类器,待检查的句子通过此分类器被分配到某个负实例特征表里进行匹配搜索,而此特征表的特征属性数和记录数都要远远小于原始负实例数据库中相应的数目，从而大大提高了检查的速度,同时不影响语法检查的精度。实验表明,提出的方法在保证语法检查精确度的同时有更快的速度。

因为 String Kernel 和 KPCA 都是新兴的理论,还远远没有达到成熟的阶段,结合 String Kernel 和 KPCA 在语法检查中的应用研究方法还会有进一步的发展。具体来说,如高维空间维数的确定、字符串长度 k 的计算、延迟因子 λ 的取值、子串跨度 l 的取定等方面,都还存在改进的空间,值得作进一步的研究,以提出更好的理论和算法。

参考文献:

[1] Atwell E,Elliott S.Dealing with ill-formed English text[M]//The Computational Analysis of English:A Corpus-Based Approach.De Longman,1987.

[2] Lukich K.Techniques for automatically correcting words in text[J].ACM Computing Surveys,1992,24(4):377-438.

[3] Peterson J L.Computer programs for detecting and correcting spelling errors[J].Communication of the ACM,1980,12:676-687.

[4] Golding A R.A window-based approach to context-sensitive spelling correction[J].Machine Learning,1999,34:107-130.

[5] 张仰森,曹元大,俞士汶.基于规则与统计相结合的中文文本自动查错模型与算法[J].中文信息学报,2006,4:1-7.

[6] 张仰森,曹元大.基于统计的纠错建议给出算法及其实现[J].计算机工程,2004,30(11):106-109.

[7] Watson I,Marir F.Case-based reasoning:A review[J].Knowledge Engineering Review,1994,9(4):355-381.

[8] 廖信海.基于实例和规则相结合的语法检查研究及系统实现[D].广州:中山大学,2003.

[9] Cristianini N,Shawe-Taylor J.An introduction to support vector machines and other kernel-based learning methods[M].New York,NY:Cambridge University Press,1999.

[10] Lodhi H,Saunders C,Shawe-Taylor J,et al.Text classification using string kernels[J].The Journal of Machine Learning Research,2002,2:419-444.

[11] Scholkopf B,Smola A,Muller K R.Nonlinear component analysis as a kernel eigenvalue problem[J].Neural Computer,1998,10:1299-1319.

[12] Vapnik V.Statistical learning theory[M].New York:Wiley,1998.

[13] Muller K R.An introduction to kernel-based learning algorithms[J].IEEE Transactions on Neural Networks,2001,12(2):181-201.

(上接 128 页)

5　结束语

Li-Yang 签名方案在一定程度上是一种效率较高的有序多重数字签名方案,它很好地实现了有序签名的目的。但是由于其在签名过程中只验证签名的有效性而忽略了对签名用户的公钥进行检验,从而造成了系统的漏洞。因此伪造者可以通过伪造签名用户的公钥,对系统进行伪造攻击。本文对原来的伪造攻击的方法进行了扩展,同时指出该方案还容易出现相邻用户之间互换签名,造成签名顺序混乱的情况。最后给出了一种基于 Li-Yang 签名方案的新的有序多重签名方案。新的签名方案能够有效抵抗 Li-Yang 签名方案中存在的伪造攻击和互换签名,并且在签名过程中,系统没有增加额外的开销,与文献[4]的改进方案相比,不仅安全性有了提高,而且减少了两次求幂运算。

参考文献:

[1] Itakura K,Nakamura K.A public key cryptosystem suitable for digital multi-signature[J].NEC Res and Develop,1983,71(10):1-8.

[2] 李子臣,杨义先.Elgamal 多重数字签名方案[J].北京邮电大学学报,1999,22:30-34.

[3] Harn L,Xu Y.Design of generalised ElGamal type digital signature schemes based on discrete logarithm[J].Electronics Letters,1994,30(24):2025-2026.

[4] 王晓明.一种多重数字签名方案的安全性分析[J].南开大学学报:自然科学版,2006,36:33-38.

[5] 杜海涛,张青坡.一个新的离散对数有序多重签名方案[J].计算机工程与应用,2007,43(2):148-150.