第35卷 第23期 计 算 机 工 程 2009年12月
Vol.35 No.23 Computer Engineering December 2009
·博士论文· 文章编号：1000—3428(2009)23—0012—03 文献标识码：A 中图分类号：TP391

String核负实例语法特征提取算法

吕 威[1,2]，林文昶[2]，李 磊[2]
(1. 北京师范大学珠海分校信息技术学院，珠海 519085；2. 中山大学软件研究所，广州 510275)

摘 要：通过String核方法把语法数据库中的负实例转化成核矩阵，采用Kmeans聚类算法对核矩阵进行聚类，将原始负实例数据库分成多个容量较小的特征数据表，使大规模 $O(n^3)$核矩阵转换为 $n/s\times O(s^3)$ ($s<<n$)矩阵，以减少运算量。分析语法检查精度随 Kmeans 聚类参数的变化规律。实验结果表明，该算法在不降低语法检查精度的前提下提高了语法检查速度。

关键词：Kmeans方法；聚类；String核；负实例；特征提取

Grammatical Feature Extraction Algorithm for String Kernel False Instance

LV Wei[1,2], LIN Wen-chang[2], LI Lei[2]
(1. School of Information Technology, Zhuhai Campus, Beijing Normal University, Zhuhai 519085;
2. Software Research Institute, Zhongshan University, Guangzhou 510275)

【Abstract】This paper translates false instance in grammatical database to kernel matrix through String kernel method, uses Kmeans clustering method to cluster the kernel matrix and separate the original false instance database into many characteristic tables with small capacitance. It transforms large scale $O(n^3)$ kernel matrix into $n/s\times O(s^3)$ ($s<<n$) matrix to decrease calculation amount, and analyzes the rule of the grammatical check accuracy with the change of Kmeans clustering parameters. Experimental results show that this algorithm can enhance the running speed without decreasing the accuracy of grammatical check.

【Key words】Kmeans method; clustering; String kernel; false instance; feature extraction

1 概述

现有语法层次错误检查以词性标注系统为基础[1]，词性标注系统对句子中单词的标记以“所要标注的句子是正确的”为前提，而词性标注系统的标注准确率并非都是100%，因此，会影响语法错误检查的准确率，甚至导致误判。文献[2]提出一种基于错误实例(负实例)[3]和错误特征相结合的方法，主要根据最相近的负实例进行查错，而不以词性标注系统为基础。该方法避免了词性标注系统带来的不确定性，可以极大减少错误判别概率，但它搜集的负实例表很大，如果每检查一个待检句子就对匹配相应的整个负实例表进行查找，将需要大量时间，导致效率很低。

本文提出使用String核[4]方法进行负实例语法特征提取，但该方法也存在运行时间过长的问题，因为它需要进行大规模核矩阵计算，矩阵规模 $O(n^3)$ 和样本数量 n 相关，所以当样本数量 n 很大时，该矩阵甚至无法求解。鉴于此，本文通过Kmeans 聚类方法[5]将原始负实例数据库分成多个容量较小的特征数据表，解决了String核方法中大规模矩阵的计算效率问题。对聚类后数据进行特征提取，为语法检查系统设计一个分类器，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。

2 算法描述

2.1 基于String核的核相关矩阵算法

String 核函数 $k(x,y)=\phi(x)\cdot\phi(y)$ 可以将低维空间的非线性问题转化成高维空间的线性问题，且只要进行点积运算而不必明确求出映射函数[6]。String核是特别针对字符设计的核函数。

String 核方法可以用来对比 2 个字符串实例的相似度，其中的核函数 K 即为对比2个字符串相似度的函数。需要用延迟因子 λ 来辅助设定不同的权值。例如，对“car”，“cat”，“bat”和“bar”等长度为3的字符串，假设子串长度 k=2，则有一个 8 维的特征空间 F，其中，$\lambda\in(0,1)$。如 $\phi(cat)=(\lambda^2,\lambda^3,\lambda^2,0,0,0,0,0)$，其中，$\lambda$ 值的指数取值 l 是子串在单词里的跨度。把跨度作为 λ 的指数可以反映单词的相关信息，连续子串的映射函数值(权值)较大，而不连续子串的信息没有丢失。

可以使用 ϕ 函数值的内积定义核函数 K 为对比2个字符串相似度的函数。如“car”和“cat”的相似度可以用 $K(car,cat)=\phi(car)\cdot\phi(cat)=\lambda^4$ 来衡量。

算法 1 核相关矩阵算法

输入 负实例表 f (设共有 n 条记录)，延迟因子 λ，子串长度 k

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

Step1 根据 k 值确定特征空间的维数 d。

Step2 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。

Step3 所有 ϕ 函数映射值构成一个 n×d 的延迟矩阵 **T**。

基金项目：国家自然科学基金资助项目(10471156, 10531040)
作者简介：吕 威(1978－)，男，讲师、博士研究生，主研方向：数据分析，机器学习；林文昶，硕士；李 磊，教授、博士生导师
收稿日期：2009-06-05 **E-mail：**luwei00@126.com

Step4 对 **T** 中任意 2 个行向量 $\mathbf{T}_i$,$\mathbf{T}_j$,$1\leqslant i\leqslant n$, $1\leqslant j\leqslant n$, $i\neq j$，计算其核函数值

$K(T_i,T_j)=T_iT_j$

2.2 基于Kmeans聚类的String核负实例语法特征提取算法

由于得出的核矩阵 $\boldsymbol{K}$ 大小为 $n\times n$，因此会碰到一个与样本数目 n 相关的问题——核方法中需要计算和存储与样本数目有关的几个 $n\times n$ 矩阵，而求解一个 $n\times n$ 矩阵的特征值问题的时间复杂度是 $O(n^3)$。一般来说，一个文档中的样本数量为几万到几十万，甚至更多，如果 n 很大，就会对 $\boldsymbol{K}$ 的求解带来很大困难。因此，本文提出一种基于 Kmeans 聚类的特征提取算法，让大规模 $n\times n$ 核矩阵 $\boldsymbol{K}$ 转变成 $t=n/s$ 个小规模 $s\times s$ 矩阵 $\boldsymbol{K}_i$，其中，$i=1,2,\cdots,t$，从而使计算规模从 $O(n^3)$ 下降到 $tO(s^3)=\frac{n}{s}O(s^3)$，极大提高了计算效率。可以证明，该聚类方法不影响语法检查的精确度。

通过 Kmeans 聚类后，每个聚类内的负实例数目不多，类的特征提取很容易实现。把特征提取出来后，可以构造出特征索引表。

算法 2 基于 Kmeans 聚类的 String 核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

输出 多个特征特征表

Step1 根据算法 1 计算核相关矩阵 **K**。

Step2 根据 Kmeans 方法将核矩阵 **K** 聚类成 $t=\frac{n}{s}$ 个矩阵 K_i，其中 $i=1,2,\cdots,t$。

Step3 根据每个 K_i，提取各个类的特征。

得到特征索引表后，通过多个输出的特征表已经设计好了一个分类器。对新来待检查文档，只要根据算法 1 计算出其 ϕ 函数映射值，就可以基于负实例推理得出其应该属于哪个错误特征表，判断其有无错误。新来子串的检查过程见图 1。

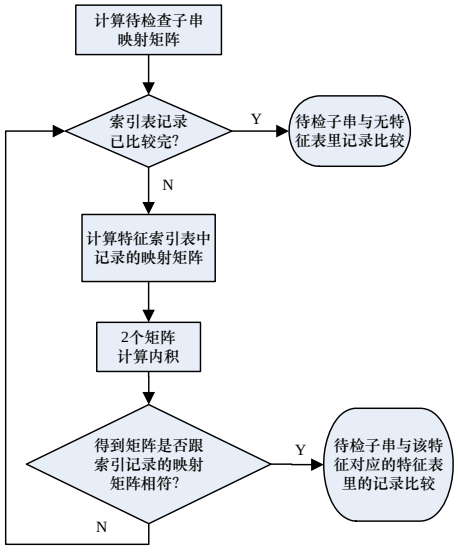

图 1 新来子串的检查过程

容易证明使用 Kmeans 聚类方法后得到的分类器准确率与原分类器准确率相同。因为本文方法只是把一个大表分成了多个小表，而这些小表合在一起完整地代表了大表的所有信息，所以准确率不会改变。

3 实验结果

在大量数据上对本文算法进行实验，结果表明其实用性较高、效果较好。下文给出对一个较有代表性的数据集的实验结果。

3.1 实验比较分析

负实例表 f 共有 3 373 条负实例，子串长度 k 取为 2，则特征空间维数为 676。取 λ =0.7，根据算法 1 可以得出一个 3 373×676 的延迟矩阵 $\boldsymbol{T}$。

例如，记录“focus can be made on”的描述如表 1 所示。

表 1 “focus can be made on”的描述

维坐标	0	…	3	…	13	…	31	…	53	…	675
对应子串	aa	…	ad	…	an	…	be	…	ca	…	zz
ϕ 映射值	0.00	…	0.49	…	0.49	…	0.49	…	0.49	…	0.00

从 $\boldsymbol{T}$ 出发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”)

该值即 2 个错误子串的相似系数，由所有相似系数组成的 3 373×3 373 的矩阵构成负实例表 f 的核相关矩阵。

假设该核矩阵计算规模很大，需要使用 Kmeans 算法对其进行聚类。若取 $t=10$，则原核矩阵转变成 10 个约 337×337 的核矩阵，计算量极大减少。且通过聚类后，在每个类中的负实例数目不多，容易在其上实现特征提取。

提出上述特征后，可以构造特征索引表，按图 1 所示流程对待检查文档进行错误检查。

实验环境为 Windows XP、1 GB 内存、CPU P4 3 GHz。由于本文算法不会降低检查精度，因此主要比较运行时间，结果如表 2 和图 2 所示，可以看出本文算法运行速度更快。

表 2 文献[2]方法和本文方法在不同数据集上的运行时间

数据集中的错误比例/(%)	文章包含的单词数	运行时间/s	
		Wordhelp[2]	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

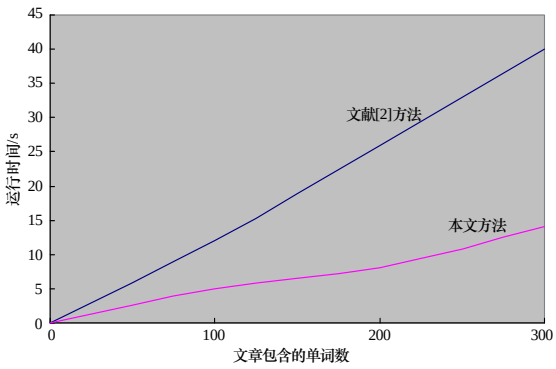

图 2 含 25%错误文章的运行时间

在不同错误比例和单词数下，本文方法的运行时间都比 Wordhelp 方法少，且随着单词数的增加，其优势越来越大。随着错误的增多，Wordhelp 方法的错误检查速度有所提高，由于对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，因此用时越少。

3.2 聚类参数对准确率的影响

聚类方法中的一些参数选择，如类的数目、每类中含有

的记录数等是一个开放问题，目前还没有一个很好的解决方法。本文给出一个实验结果，显示聚类数目和每类中记录数的变化对准确率的影响，从而确定较合适的聚类参数。

假定聚类后得到的某个类含有 y 个记录，其中，有 x 个记录不应该属于 y，则该类的准确率为

$$e=\frac{y-x}{y}\times 100\%$$

如果共分为 n 类(如 50 类)，每个类的准确率为 e_i (i=1,2,⋯,n)，且每个类含有的记录总数为 y_i (i=1,2,⋯,n)，总记录数目为 Y，则所有类的加权准确率为

$$E=(\sum_{i=1}^{n}(\frac{y_i}{Y}\times e_i))\times 100\%$$

表 3 描述了聚类方法中的参数选择对准确率的影响，其中，大类表示包含 100 个以上记录的类；中类表示包含 10 个~99 个记录的类；小类表示包含 1 个~9 个记录的类。

表 3　聚类方法中的参数选择对准确率的影响　(%)

λ	分类大小	聚为 30 类	聚为 50 类	聚为 100 类
0.5	大类	83.7	80.7	76.6
	中类	70.3	74.1	64.3
	小类	62.7	66.8	67.3
0.7	大类	85.1	83.2	78.5
	中类	71.4	76.8	65.1
	小类	61.4	65.2	68.2

由表 3 可知，当延迟因子 $\lambda=0.7$ 时，分成大类且聚为 30 类时准确率最高，在后继实验中可以参考该值进行取值。

4　结束语

本文通过聚类方法加速核矩阵的计算速度，其效果较好。String 核和 Kmeans 算法在不断发展，因此，如何更好地将两者相结合应用于语法检查有待进一步研究，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 的取值等。

参考文献

[1] Golding A R. A Window-based Approach to Context-sensitive Spelling Correction[J]. Machine Learning, 1999, 34(1-3): 107-130.

[2] 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.

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

[4] 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.

[5] Macqueen J. Some Methods for Classification and Analysis of Multivariate Observations[C]//Proc. of the 5th Berkeley Symp. on Math. Statist. Berkeley, CA, USA: University of California Berkeley Press, 1967: 281-297.

[6] Muller K R. An Introduction to Kernel-based Learning Algorithms[J]. IEEE Transactions on Neural Networks, 2001, 12(2): 181-201.

编辑　陈　晖

（上接第 11 页）

群大小为 10 000 个体，最大演化代数为 1 000 代。将 FPGA 中“双输入与非门”作为组织功能电路的基本单元，搭建二叉树型电路结构。电机转子位置信号反馈端口 S_0, S_1, S_2 作为二叉树的叶子节点，由若干个与非门相互连接构成 6 个二叉树，FPGA 对驱动模块中的各个功率管的控制端口 M_0~M_5 分别作为 6 个二叉树的各自根节点。对转子位置信号反馈端口、FPGA 控制端口、与非门的连接关系进行编码，形成 EHW 中的染色体。

完成种群初始化后，对种群中的每个染色体进行适应度评估。设定电机换相表(表 2)作为适应度函数，令各种 S_0, S_1, S_2 组合输入功能模块后，产生输出 M_0~M_5，利用适应度函数得出适应度值并进行累计。针对不同输入，若所有输出均与表 2 一致，则演化成功，此功能模块恢复工作，可以重新组成 TMR 结构。否则，对种群继续演化，根据适应度值进行选择、交叉、变异等操作，生成新种群，直到演化成功或达到了最大演化代数的设定值(此时可以重新初始化种群并再次进行演化)。

当演化生成符合要求的个体后，对其染色体进行解码，形成对应的 VHDL 代码，利用集成编译环境生成相应配置文件并下载至 FPGA 中。此时，重新形成了该模块的电机控制电路的结构和功能，恢复了系统的 TMR 结构。

恢复后的功能模块与其他功能模块相比，尽管功能一致，但具体的片内单元连接方式等方面可能存在差异，并非先前电路在结构上的完全复制。因此，在进化、下载过程中可能自行“绕过”故障芯片内的受损单元而恢复原有功能。这种效果是使用常规冗余和备份方法所难以获得的。

3　结束语

TMR-EHW 机制融合了 TMR 与 EHW 技术各自的优势，采用 TMR 结构快速发现并定位故障，保证系统中某一功能模块故障后仍然能够正常输出，至少可为后续的修复工作赢得足够的处理时间。利用 EHW 技术可使故障模块的功能通过演化而得到部分或完全恢复，使系统能从容错运行状态恢复到正常运行状态。而结合 FPGA 的“结构硬化”方式使电机控制系统可靠性得到进一步加强，降低了人工参与的要求，初步实现了系统自维护的设想。该技术在部分环境参数和条件无法确定或预测的宇航、深潜等恶劣环境中具有较好的应用前景，并有望进一步拓展至其他可靠性要求较高的领域。

参考文献

[1] 吴彩华. 遗传算法在演化硬件技术中的应用研究[D]. 石家庄: 军械工程学院, 2005.

[2] 潘正君, 康立山, 陈毓屏. 演化计算[M]. 北京: 清华大学出版社, 1998.

[3] 康立山, 何　巍, 陈毓屏. 用函数型可编程器件实现演化硬件[J]. 计算机学报, 1999, 22(7): 781-784.

[4] 吴会丛. 基于 EHW 的芯片级电磁损伤自修复技术研究[D]. 石家庄: 军械工程学院, 2006.

[5] 原　亮, 丁国良, 刘文冰, 等. TMR 整体硬化技术及其在电控系统中的应用[J]. 计算机工程, 2006, 32(21): 249-251.

编辑　陈　晖