Web表格定位技术的研究与实现

廖 涛[1,2] 刘宗田[2] 孙 荣[2]

(安徽理工大学计算机科学与工程学院 淮南 232001)[1] (上海大学计算机工程与科学学院 上海 200072)[2]

摘 要 Web表格的定位作为Web表格抽取的一个重要研究内容,现在越来越得到更多人的重视。根据Web表格的结构标记和自定义的启发式规则,通过对〈TABLE〉嵌套问题的解决、数据表格完整性的判断、〈TABLE〉树的遍历来完成表格的定位。

关键词 DOM树,表格定位,启发式规则,〈TABLE〉嵌套,遍历

中图法分类号 TP311 **文献标识码** A

Research and Implementation of Web Table Positioning Technology

LIAO Tao[1,2] LIU Zong-tian[2] SUN Rong[2]

(Department of Computer Science and Engineering, Anhui University of Science & Technology, Huainan 232001, China)[1]

(School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China)[2]

Abstract Web table positioning technology is considered as essential components of Web table information extraction, and more and more people pay attention to them. This paper realized table positioning according to Web table structure label and heuristic method rules of user-definition, which includes the solution of 〈TABLE〉 nesting problem, the determination of table data's integrality, and traversal of 〈TABLE〉 tree.

Keywords DOM tree, Table positioning, Heuristic method rules, 〈TABLE〉 nesting, Traversal

1 引言

随着Internet的迅速发展,我们真正迎来了信息爆炸时代。目前各类信息服务网站提供了大量的信息资源,而在大量的网页资源中,Web表格是极其重要而又规律的,现在表格(Table)作为一种重要的信息表现形式已广泛地应用于Web网页中。

Web表格抽取任务的提出始于上世纪90年代末[1,2],主要研究Web页面上的表格,包括表格定位、表格结构与内容分析、提取表格中有价值的数据信息以及表格合并等。本文主要研究Web表格的定位。

Web表格的定位是Web表格抽取任务中的一个重要内容,指从Web页内找到表格区域,并去除"假表格"等噪音。真假表格的判断需要构造分类器,目前主要有3种方式:

1)基于机器学习分类——需要选取表格特征信息以及样本集来训练分类器;

2)基于人工构造规则分类——需要构造表格特征的启发式规则;

3)基于本体辅助分类——在前两种方式的基础上,在限定领域内利用本体判断真假表格。

国外关于Web表格定位的研究中,Hurst归纳了Web表格的两类特征,即DOM特征(5个)和几何模型特征(3个),并利用两种训练算法,即贝叶斯(Naive Bayes)和甄别(Winnow)对Web表格进行特征训练[3]。Wang和Hu提出了Web表格定位时需考虑的3类特征:布局特征、内容类型特征和词组特征,用基于决策树学习方法和基于SVM学习方法的分类算法实现表格定位[4]。BYU研究小组的Cui Tao将Web表格分为顶层表格(Top-LevelTables)和链接页面表格(Linked- PageTables),其基于页面训练集分别提取相应的表格特征,构造启发式规则,并引入领域本体对真假表格进行判断[5]。

国内台湾学者H. Chen等人提出识别Web数据表格的两条规则[6]:至少含有两个单元格以表示属性和值。表格中含有许多超链接、表单和图像的区域,被视为非数据表格区域。在此基础上,利用表格单元格的3种相似度(字符串相似度、命名实体相似度和数值类型相似度)与阈值的比较来进一步过滤非目标表格。林科锵、林琳在BYU研究小组的研究基础上,将表格处理过程分解为表格的定位、表格结构识别以及表格内容抽取3个步骤,并给出一个基于本体的通用Web表格信息抽取系统模型[7,8]。

本文对上面的这些Web表格定位方法进行研究,发现基于机器学习分类方法的效果是最好的,但对于发生变化的网页或没有学习归纳过的网页格式,需要重新学习;使用基于领域本体的方法一般只能针对限定领域,并且需要领域专家来创建某一应用领域的本体,工作量大且不能直接用于其他领域。

到稿日期:2008-10-21 返修日期:2009-01-07 本文受国家自然科学基金(60575035),上海市重点学科建设项目(J50103)资助。

廖 涛(1977-),男,博士研究生,讲师,主要研究领域为Web数据挖掘、文本分类等,E-mail:hntliao@netease.com;刘宗田(1946-),男,教授,博士生导师,主要研究领域为人工智能和软件工程等;孙 荣(1977-),男,硕士研究生,主要研究领域为数据挖掘、自然语言处理等。

2 Web 表格的定位

本文在现有表格定位方法的基础上，提出根据表格的结构标记，通过定义一些识别表格特征的启发式规则来实现 Web 表格的定位。Web 页面中的所有元素用 DOM 树表示，这里我们只关心树中的 TABLE 结点。

2.1 两个需要解决的问题

在表格定位的实现过程中，本文发现需要解决好两个问题：数据表格的判断和〈TABLE〉嵌套的问题。

2.1.1 数据表格的判断

在网页中 Web 表格是介于标记〈TABLE〉和〈/TABLE〉之间的内容。在表格的定位过程中，〈TABLE〉标记是识别表格的一个重要依据。但是并不是每个〈TABLE〉标记的存在都能确定一个真正的数据表格的存在。这里所说的数据表格是指一类用来组织和显示丰富数据信息的〈TABLE〉区域，它具有简洁、清晰、逻辑性和对比性强等特点。据统计，在一个特定领域中，Web 表格是真数据表格的数量在 30%以下[9]。

非数据表格是指被用来进行页面布局的〈TABLE〉区域，其中可能包含了很多的图片、纯文字和超链接的信息，这些都属于噪音信息，本文称之为“假表格”。

在对大量的 Web 页面进行观察后，本文得到了一些关于判断数据表格的启发式规则。

规则 1　如果〈TABLE〉标记中包含〈TH〉或〈CAPTION〉标记，则该表格是一个数据表格。

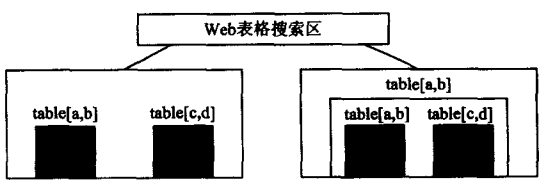
规则 2　如果〈TABLE〉标记区域中包含大量的图片、框架、表单、脚本标记，则该表格为非数据表格。

规则 3　如果〈TABLE〉中的单元数太少，则该表格为非数据表格。

规则 4　如果〈TABLE〉中的空单元的数量占到了总单元数的一半，则该表格为非数据表格。

2.1.2 〈TABLE〉嵌套问题的处理

表格定位识别时可能遇到的〈TABLE〉标记的两种不同情况，如图 1(a)、(b)所示。



(a)非嵌套的情况　(b)嵌套的情况

图 1　表格定位识别的两种情况

对于图 1(a)中非嵌套的情况，本文根据数据表格的判定规则，能够很容易地对表格进行定位识别。但对于图 1(b)中〈TABLE〉嵌套的情况，则需要通过对表格中的内容进行识别来完成表格的定位。

本文通过观察发现，如果 Web 文本出现了〈TABLE〉嵌套的情况，一般会有以下 3 种可能：

(1)最内层〈TABLE〉标记区域中的表格是一个“完整”的数据表格。在很多的〈TABLE〉嵌套中，外层的〈TABLE〉都是用来控制页面布局的，只有里层的〈TABLE〉才是真正的数据表格。

(2)最内层〈TABLE〉标记区域中的表格是一个“假表格”。在 Web 文本中经常会出现使用〈TABLE〉嵌套来控制页面布局的情况。

(3)最内层〈TABLE〉标记区域中的表格是一个“非完整”的数据表格，它所包含的内容只是某个 Web 表格中的部分内容。

文献[3-8]中对于多重〈TABLE〉嵌套的处理，要么只是简单地针对最里层的〈TABLE〉进行分析，忽略了在一个“完整”数据表格中可能会出现多重〈TABLE〉嵌套的情况；要么就是需要根据领域知识或相关词汇进行判断，这样的方法需要由领域专家来创建某一应用领域的本体，工作量大且不能直接用于其他领域。

本文对表格定位中的多重〈TABLE〉嵌套情况的处理如下：

(1)如果出现〈TABLE〉嵌套，找到最里层的〈TABLE〉。如果它是一个“假表格”，则表明这里的多重〈TABLE〉嵌套只是用于控制页面布局，其中并不包含本文中所需要的数据表格。

(2)如果最里层的〈TABLE〉是一个“完整”的数据表格，则它外层的〈TABLE〉均属于非数据表格。

(3)如果最里层的〈TABLE〉是一个“非完整”的数据表格，表格区域需要向外层〈TABLE〉扩展，直到找到一个“完整”的数据表格。该“完整”数据表格的外层〈TABLE〉均属于非数据表格。

对于“完整”的概念，主要针对有时存在这样一种情况，即在一个数据表格中也可能会出现多层〈TABLE〉嵌套，如图 2 所示。这是网易证券页面上的一个 Web 表格，HTML 简化源码如图 3 所示。

名称	最近价	涨跌幅%	成交量
皖维高新	4.68	10.12	37664
金瑞矿业	5.02	10.09	86544
现代制药	6.33	10.09	257312
上海贝岭	9.41	10.06	60159
东方通信	4.71	10.05	546234
法拉电子	13.48	10.04	37824
三精制药	11.62	10.04	48085
申通地铁	6.04	10.02	144234
恒生电子	9.45	10.01	80530
皖维股份	4.84	10.00	501162

图 2　实例 Web 表格页面

```
〈TABLE〉
  〈TR〉
    〈TD〉上海 A 股涨幅排名〈/TD〉〈/TR〉
  〈TR〉
    〈TD〉名称　最近价　涨跌幅%　成交量〈/TD〉〈/TR〉
  〈TR〉
    〈TD〉
      〈TABLE〉
        〈TR〉
          〈TD〉皖维高新〈/TD〉
          〈TD〉4.68〈/TD〉
          〈TD〉10.12〈/TD〉
          〈TD〉37664〈/TD〉〈/TR〉
        〈TR〉
          ……
      〈/TR〉〈/TABLE〉〈/TD〉〈/TR〉〈/TABLE〉
```

图 3　实例 Web 表格页面的简化源码

从图 2 和图 3 中可以看出，最里层的〈TABLE〉所包含的并不是“完整”的数据表格，而只是数据表格的一部分。

本文中的数据表格“完整性”的判断，主要依据自定义的两条启发式规则：

(1)如果〈TABLE〉中前几行的单元类型过于一致，则可初步判定该〈TABLE〉中可能包含的是“非完整”的数据表格。

一般来说，数据表格的最上面一行应该是头元素行(属性行)，该行单元信息类型基本上都是字符型，它和下面数据行(值行)中单元信息类型(一般为数字类型)应该是不太一致的。

在图 3 中，最里层〈TABLE〉中所包括的行全部都是数据行，它们对应单元的类型是一致的。

(2)当扩展到外一层的〈TABLE〉时，并没有引入大量的图片、框架、表单、脚本标记，则可以认为通过向外扩展增加了数据表格的完整性。

在上面的两条规则同时满足的情况下，才可以判定当前的〈TABLE〉中包含的是“非完整”的数据表格，允许扩展到外一层的〈TABLE〉，然后再对数据表格的完整性进行判断。

2.2 〈TABLE〉树

图 4 为网易证券页面截图。在该页面源码中包含数十个〈TABLE〉标记，其中多重〈TABLE〉嵌套的情况很多。为了能够准确地完成表格的定位，本文根据 Web 文本源码中〈TABLE〉结构关系构造出了〈TABLE〉树，并根据上面提出的〈TABLE〉嵌套解决思想，通过对〈TABLE〉树进行一次后序遍历，从而找到每一个完整的数据表格。

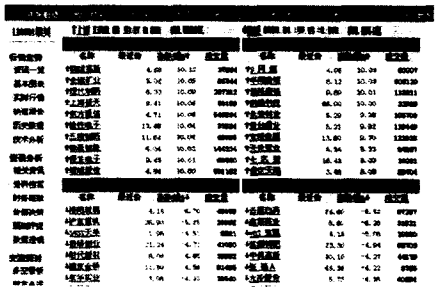


图 4 网页截图

1)〈TABLE〉树的建立

假设某 Web 文本源码中〈TABLE〉结构关系如图 5 所示。

```
〈TABLE〉
    〈TABLE〉
        〈TABLE〉
        〈/TABLE〉
    〈/TABLE〉
〈/TABLE〉
〈TABLE〉
    〈TABLE〉
        〈TABLE〉
        〈/TABLE〉
        〈TABLE〉
        〈/TABLE〉
    〈/TABLE〉
    〈TABLE〉
        〈TABLE〉
```

```
            〈TABLE〉
                〈TABLE〉
                〈/TABLE〉
                〈TABLE〉
                〈/TABLE〉
            〈/TABLE〉
        〈/TABLE〉
        〈TABLE〉
            〈TABLE〉
            〈/TABLE〉
        〈/TABLE〉
    〈/TABLE〉
〈/TABLE〉
```

图 5 〈TABLE〉结构关系图

根据图 5 中的〈TABLE〉结构关系，可以得到如图 6 所示的两个〈TABLE〉子树。

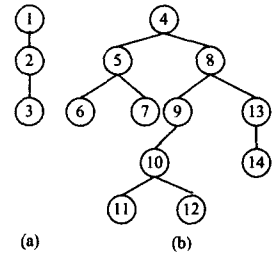


图 6 〈TABLE〉树

2)〈TABLE〉树的遍历

对于图 6(b)的〈TABLE〉树，后序遍历访问树中的每个节点。遍历过程如图 7 所示。

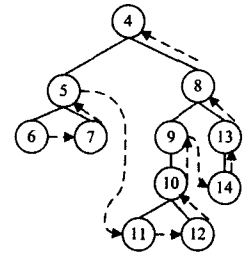


图 7 后序遍历〈TABLE〉树

对〈TABLE〉树遍历的主要目的是找到完整的数据表格。本文给每个结点定义标志属性 FLAG，并设初始值为 0。

后序遍历是从最左边的叶子结点开始的。当访问到每一个结点时，首先判断它是不是一个数据表格，这时结点标志属性 FLAG 的值将可能发生改变。

算法核心思想：

(1)如果当前结点是叶子结点

①如果结点是“假表格”，则 FLAG 值都变为－1，该结点所有祖先结点的 FLAG 值都变为－1，访问下一个结点；

②如果结点是数据表格，判断其完整性。如果“完整”，则 FLAG 值都变为 1，该结点所有祖先结点的 FLAG 值都变为－1；如果“不完整”，则 FLAG 值不变，访问下一个结点。

(2)如果当前结点不是叶子结点

①如果其 FLAG 值为 0，则判断其完整性。如果“完整”，则 FLAG 值变为 1，该结点所有祖先结点的 FLAG 值变为－

1;如果“不完整”,则 FLAG 值不变,访问下一个结点;

②如果其 FLAG 值为-1,访问下一个结点。

遍历结束后,标志属性 FLAG 的值为 1 的结点即为在 Web 文本中找到的“完整”数据表格。

3 实验结果与分析

衡量实验效果主要根据两个评价指标[10]:召回率(Recall,简写 R)和准确率(Precision,简写 P)。

召回率等于系统正确定位的表格占所有正确数据表格的比例:

$$R=\frac{\text{系统正确定位的表格}}{\text{所有应该得到的正确结果}}\times 100\%$$

准确率等于系统正确定位的表格占系统所得到结果的比例:

$$P=\frac{\text{系统正确定位的表格}}{\text{系统得到的结果}}\times 100\%$$

一般来说,召回率和准确率存在相互影响。对于同一次表格定位的结果,随着召回率的提高,准确率呈下降趋势;反之亦然。

为了综合评价系统的性能,通常还需计算召回率和准确率的加权几何平均值,即 F 指数,它的计算公式如下:

$$F=\frac{(\beta^2+1.0)\times P\times R}{\beta^2\times P+R}\times 100\%$$

其中,β 是召回率和准确率的相对权重。β 等于 1 时,二者同样重要;β 大于 1 时,准确率更重要一些;β 小于 1 时,召回率更重要一些。β 取值一般为 1,1/2,2。

本文把从网易、新浪、搜狐网站中选取的 50 个包含 Web 表格的不同类型的网页作为实验样本(包含数据表格 226 个),得到的实验结果如表 1 所列。

表 1 实验结果分析

Web 数据表格的个数	召回率 R(%)	准确率 P(%)	F 指数 (%)		
			β=1	β=1/2	β=2
226	94.4	93.7	94	93.8	94.3

通过实验,可以看到使用本文方法实现 Web 表格定位的结果还是比较理想的。

此外本文将实验结果与国内外其他一些 Web 表格定位的研究成果进行对比,其结果如表 2 所列。

表 2 本文实验结果与其他研究结果比较

作者	实验数据	实验结果		
		准确率	召回率	F 指数 β=1
Chen et al[6]	918 个 YAHOO 网旅游相关表格	92.9%	80.1%	86.5%
Penn et al[11]	75 个关于电视、无线领域页面	86.3%	89.8%	88.1%
Hurst[3]	339 个任意页面的表格	95.0%	93.5%	94.2%
林科锵[7]	325 个页面 4025 个表格	91.7%	87.8%	89.7%
本文	50 个网页(226 个数据表格)	94.4%	93.7%	94.0%

从表 2 可以看出,Hurst 利用机器学习方法定位数据表格的能力更好,但该方法在时间上要求较多。本文提出的基于表格结构分析的方法也达到了较好的效果。

结束语 本文提出的 Web 表格定位方法,定义了判断完整数据表格的启发式规则,通过对〈TABLE〉树进行后序遍历,解决了多重〈TABLE〉嵌套的问题,较好地实现了 Web 表格的定位。但同时也看到,本文的 Web 表格定位方法是基于结构的,它的定位效果与 Web 网页的结构有密切关系。为了能适用各种不同的网页结构,在启发式规则的定义上还有需要改进的地方,这也是今后需要不断改进完善的方面。

参 考 文 献

[1] Hammer J, Garcia-Molina H, Cho J, et al. Extracting semi-structured information from the Web[J]. SIGOD Record, 1997, 26(2): 18-25

[2] Lim S, Ng Y. An automated approach for retrieving heirarchical data from HTML tables[A]// Proceedings of the 8th International Conference on Information and Knowledge Management (CIKM'99)[C]. 1999: 466-474

[3] Hurst M. Classifying Table Elements in HTML [A] // Proc. The 11th International World Wide Web Conference[C]. WWW 2002, Sheraton Waikiki Honolulu, Hawaii, USA, May 2002. http://www2002.org/CDROM/poster/115/index.html

[4] Wang Y, Hu J. A Machine Learning-based Approach for Table Detection on the Web[A]// Proceedings of the 11th International Conference on WWW[C]. 2002: 242-250

[5] Cui Tao. Schema Matching and Data Extraction over HTML Tables[D]. USA: Brigham Young University, 2003

[6] Chen H, et al. Mining Tables from Large Scale HTML Texts [A]// Proceedings of the 18th International Conference on Computational Linguistics[C]. 2000: 166-172

[7] 林科锵. Web 页中表格结构识别的研究与实现[D]. 成都:电子科技大学,2006

[8] 林琳. 基于 Ontology 的 Web 表格内容抽取的研究与实现[D]. 成都:电子科技大学,2006

[9] Chen Hsin-Hsi, Tsai Shih-Chung, Tsai Jin-He. Mining tables from large scale html texts[A]// The 18th International Conference on Computational Linguistics[C]. July 2000: 166-172

[10] Robert G, Wilks Y. Information extraction: Beyond document retrieval[J]. Journal of Documentation, 1998, 54(1): 70-105

[11] Penn G, Hu J, Luo H, et al. Flexible Web document analysis for delivery to narrow-band width devices[A]// Proceeding of the 5th International Conference on Document Analysis and Recognition(ICDAR)[C]. Seattle, USA, 2001: 1074-1078