Deep Web 查询接口的自动判定

高 岭,赵朋朋,崔志明
(苏州大学 智能信息处理及应用研究所,江苏 苏州 215006)

摘 要:传统搜索引擎仅可以索引浅层 Web 页面,然而在网络深处隐含着大量、高质量的信息,传统搜索引擎由于技术原因不能索引这些被称之为 Deep Web 的页面。由于查询接口是 Deep Web 的唯一入口,因此要获取 Deep Web 信息就需判定哪些网页表单是 Deep Web 查询接口。文中介绍了一种利用朴素贝叶斯分类算法自动判定网页表单是否为 Deep Web 查询接口的方法,并实验验证了该方法的有效性。

关键词:深网;网页表单;特征提取;朴素贝叶斯分类
中图分类号:TP181 **文献标识码**:A **文章编号**:1673-629X(2007)05-0148-04

Automatic Judgment of Deep Web Query Interfaces

GAO Ling, ZHAO Peng-peng, CUI Zhi-ming
(Institute of Intelligent Information Processing and Application, Suzhou University, Suzhou 215006, China)

Abstract: Traditional Web search engines work well for finding crawlable pages, but they ignore the tremendous amount information hidden behind query forms, in large searchable electronic databases. For obtaining dynamic information, firstly query interfaces must be extracted from massive Web forms to find the entrance to the datasets. This paper describes a technique for detecting query interface using naive Bayes classification and the test results are reported.

Key words: Deep Web; HTML form; feature extraction; naive Bayes classification

0 引 言

随着 Web 数据库的广泛应用,Web 正在加速地"深化"[1]。Internet 上有大量页面是由后台数据库动态产生,现有的搜索引擎不能索引这部分页面信息,使得这部分信息对用户来说是隐藏的,称之为 Deep Web(又称为 Invisible Web, Hidden Web)。Deep Web 是一个与 Surface Web 相对应的概念,最初由 Dr. Jill Ellsworth 于 1994 年提出,指那些由普通搜索引擎难以发现其信息内容的 Web 页面[2]。2001 年,Christ Sherman, Gary Price 对 Deep Web 定义为:虽然通过互联网可以获取,但普通搜索引擎由于受技术限制而不能或不作索引的那些文本页、文件或其它通常是高质量、权威的信息[3]。

根据 BrightPlanet 公司 2000 年 3 月 13 日至 30 日搜集的数据[4],他们得到如下结果:Deep Web 的公共信息是 Surface Web 的 400~550 倍(Deep Web 的容量有 7500TB,而 Surface Web 只有 19TB;Deep Web 有近 5500 亿个独立文件,而 Surface Web 只有 10 亿);目前存在的 Deep Web 网站已经突破 20 万个;60 个最大的 Deep Web 网站共包含 750TB 的信息,比 Surface Web 信息的 40 倍还多;Deep Web 的月流量通常比 Surface Web 要多出 50%,但是 Deep Web 并不被公共互联网搜索领域所熟知;在内容上,Deep Web 网站比 Surface Web 网站要更专、更深;Deep Web 内容的全部价值是 Surface Web 的 1000 至 2000 倍;Deep Web 的信息内容往往与市场、领域和信息需求高度相关;一半以上的 Deep Web 内容存储在主题数据库中;95% 的 Deep Web 信息无需付费或订阅,用户可以直接获取。

尽管 Deep Web 信息量大、信息质量高,但由于技术原因不被主流搜索引擎所索引。搜索引擎的爬虫程序(Crawler)往往可以很容易找到数据库的查询接口页面(由于在互联网上大多数的查询接口都以 HTML 语言编写的 Form 网页表单样式出现,因此文中将忽略其它例如 JAVA 图形用户界面的查询接口)。但是搜索引擎的爬虫程序无法像人一样完成诸如查询接口

收稿日期:2006-07-05
基金项目:教育部科研重点项目(205059);教育部"高校博士学科点科研基金项目"(20040285016);江苏省高技术研究计划项目(BG2005019)
作者简介:高 岭(1982-),男,浙江义乌人,硕士研究生,研究方向为 Web 数据挖掘、个性化服务技术;崔志明,教授,博士生导师,研究方向为智能化信息处理、计算机网络应用与数据库。

填写这样的动作,即无法向数据库提交查询,与数据库进行交互,因此也就无法抓取到隐藏在数据库查询接口后面的丰富信息。

查询接口是 Deep Web 后台数据库的唯一入口,因此如何判定哪些网页表单是查询接口对 Deep Web 信息获取至关重要。文中介绍如何通过网页表单的 HTML 结构信息进行特征提取,并使用自动分类方法将各种网页表单进行分类,从而获得 Deep Web 查询接口。

1　相关研究工作

目前已有一些关于 Deep Web 信息获取方面的研究,然而大部分的工作都集中在网页表单自动填写、数据源选择等方向。这些工作都是在已获取 Deep Web 查询接口的基础上完成的,但是很少有关于如何判定某网页表单是 Deep Web 查询接口方面的研究。

研究比较广泛的是由 Juliano Palmieri Lage 等人提出的判定方法[5],这种方法用了两个根据实际经验总结出来的规则来判断网页表单是否为查询接口,虽然可以获取大量的查询接口,但是不具备自动学习的功能,具有一定的局限性,效果并不十分理想。

而一些已有的 Deep Web 分类目录搜索引擎如 Completeplanet, InvisibleWeb 虽然可以自动或半自动地对 Deep Web 进行分类,但并没有公开其技术细节。

2　查询接口的判定

网上可检索的动态信息大部分是结构化的数据,这些信息存储在关系数据库系统中,隐藏在查询接口的后面。当需要检索数据时,必须使用网站的查询接口进行查询,在交互式查询接口中输入查询条件,然后提交查询,最终数据库响应查询请求后,将匹配的查询结果按一定的排序规则显示给用户。

一般网络上允许用户对信息进行搜索的 Deep Web 查询接口包括:对出售商品的查询表单,对书籍目录的查询表单,对地理位置信息的查询表单等。而一些例如邮件订阅的表单,在线商店的购买表单,基于 Web 的邮件表单,还有网站的会员登录表单都不属于 Deep Web 查询接口。

通过网络爬虫程序可以抓取大量的网页,对这些已抓取的网页进行页面抽取,便可以获取网页表单结构。然而海量的网页表单,用人工的方法判断一个网页表单是否为 Deep Web 查询接口,显然是件费时费力的工作,因此必须采取一种自动的机器学习方法用以判断网页表单的类型。通常有两类方法可以将网页表单自动分成是或者不是查询接口两类。一类是提交查询法:提交一个或多个查询,根据结果页面来对网页表单进行分类;另一类非提交查询法则是直接利用网页表单的结构信息进行特征提取,从而进行分类。由于网页表单很容易获取,没有提交查询获取数据库内部数据那么复杂,并且仅是为了分类的目的而提交大量的查询则有些浪费网络和服务器资源,因此文中选择了非提交查询的方法。

由于是根据网页表单的特征进行分类的,所以用到的网页表单分类技术是文本分类技术的一种扩展,但是网页表单特征比较复杂,因此对网页表单的分类问题相对文本分类来说要更加难处理,要考虑更多的因素。能否合理地利用网页表单的各种信息,将影响到分类的效果。

图 1 是进行网页表单分类的过程示例,分为特征提取和机器学习两大关键步骤。

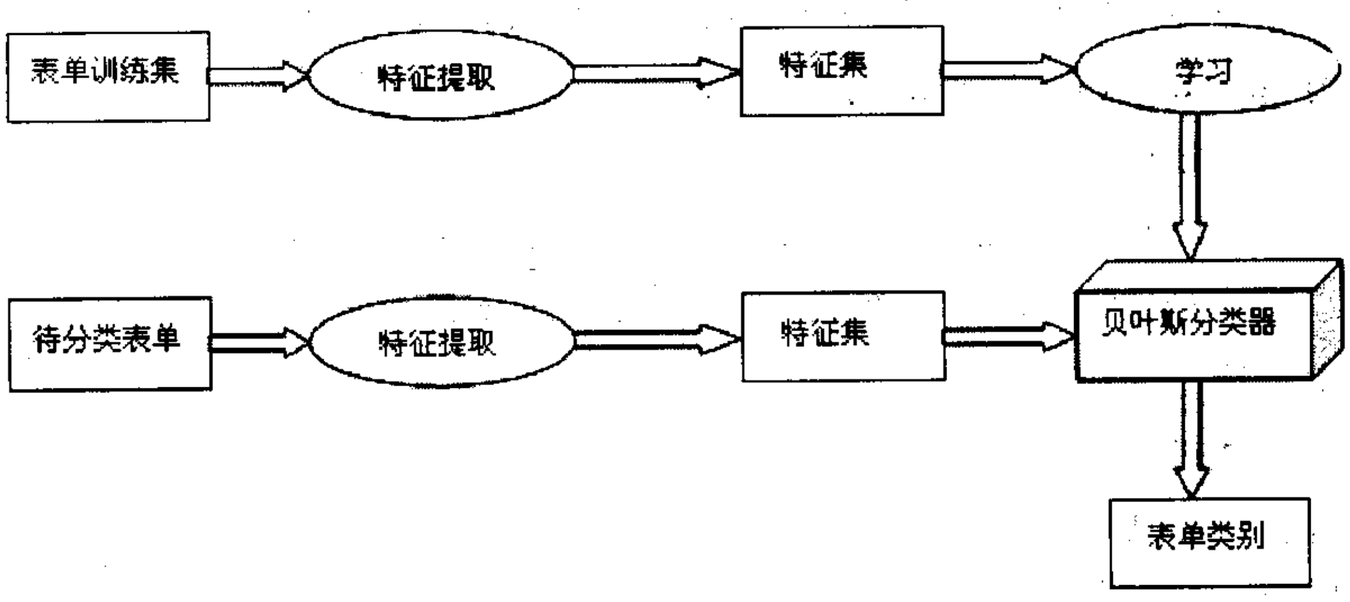

图 1　网页表单自动分类过程

2.1　网页表单特征的提取

特征提取是自动分类的前提,它可以从以下几个方面提高自动分类系统的性能。首先是分类速度,通过特征的选择,可以大大减少特征集合中的特征数,从而提高自动分类系统的运行速度,使之能够满足现实需求。二是通过适当的特征选择,不但不会降低系统的准确性,反而会使系统的精度提高。

用 HTML 表示的网页表单包含了复杂的信息结构,可以从中获取大量的有用信息集合。如何选择合适的特征对网页表单进行描述,以便有效地对网页表单自动分类,成为首先需要解决的问题。

网页表单即包含在 HTML 标签＜FORM＞…＜/FORM＞之内的相关内容。其一般格式为＜FORM action＝“URL”method＝“GET|POST”＞网页表单内容＜/FORM＞,其中参数 action 用以指明要将网页表单数据提交给的服务器处理程序(包括网络路径或相对路径)。参数 method 用以指明表示用户输入的网页表单内容以何种方式传给服务器。

网页表单的内部控件可以分三大类：INPUT 控件，SELECT 控件和 TEXTAREA 控件。INPUT 控件定义了网页表单内可以输入编辑的区域，INPUT 控件的 TYPE 属性描述了控件的类型，有 text，checkbox，radio，submit，reset，radio，image，hidden 八种，其中 submit 类型控件用于将填好的网页表单内容提交；INPUT 控件的 name 属性指定每一个控件一个名称，这个名称与控件是一一对应的，在一个网页表单中也是唯一的。服务器就是通过调用某一控件的 name 属性来获得该区域的数据。INPUT 控件的 value 属性是另一个公共属性，它可用来指定控件的缺省值。SELECT 控件用来创建一个下拉列表框或可以复选的列表框。TEXTAREA 控件用来创建一个可以输入多行的文本框。

网页表单拥有如此多的有用信息，因此可以通过提取网页表单中出现过的各种控件信息对网页表单进行特征表示，如对控件类型的提取和对控件名称的提取等。而一些用于体现控件外观表现方式的属性可以剔除，如 size，length 等。另外对于 SELECT 和 TEXTAREA 控件，由于笔者考虑的是如何通过网页表单的结构获取特征，进而得到网页表单是否为查询接口的问题，因此对下拉列表框中出现的选项内容并不做提取。

此外，还可以通过对网页表单中控件出现的次数进行特征提取，例如一个网页表单可能存在一个文本框，也可能存在多个文本框。注意到一些网站提供的高级检索往往具有多个文本框，而一些站内检索往往仅仅只有一个文本框，由此可见这对网页表单的特征提取也是有积极意义的。

综合上述观点，根据网页表单的结构，将提取以下网页表单特征：

1)网页表单＜FORM＞标签中的 name 属性值。

2)从网页表单＜FORM＞标签的 action 属性值中提取的词。

3)网页表单中出现的控件类型。

4)INPUT 控件的 name 属性值和 value 属性值。

5)SELECT 控件和 TEXTAREA 控件的 name 属性值。

6)存在于控件标签之间的词。

其中从网页表单＜FORM＞标记的 action 属性中提取的词是指将 action 属性中出现的 URL 字符串根据斜线(/)划分出来的子串。另外由于存在于网页表单控件标签之间，呈现在网页上以提示用户如何填写网页表单的词对分类也起到一定的作用，因此也将这些词作为特征提取出来。

经过上述的特征提取过程，已经可以获取大量的网页表单特征，但是网页格式灵活，可以多种格式并存，而且同一格式的网页也存在多个标准；同时网页设计人员的写作风格、习惯不尽相同。这使得提取出来的网页表单特征显得过于杂乱。因此在特征提取时，需要对特征进行标准化以提高分类的准确性。

文中根据以下几个规则对特征进行标准化：

(1)将特征中出现的英文字母统一转化为小写。

(2)去除出现在括号中的内容。

(3)如果特征中出现的内容由多个词组成，则要进行分词，去除停用词。

(4)转换缩写和简写并利用 WordNet[5]来得到统一的特征表示。

为了描述以上自动提取网页表单特征的过程，用图 2 和图 3 做了举例，图 2 中的内容是网页表单的 HTML 代码，图 3 中的内容是从中自动提取出来的特征。

```
<FORM action="user/getbook.asp" method="post" name="search">
        图书类别:
        <SELECT name="type">
        <OPTION value="c " selected>计算机</OPTION>
        <OPTION value="w">外语</OPTION>
        </SELECT>
        书名:  <INPUT  type="text"  name="title">
        作者:  <INPUT type="text" name="author">
        出版社: <INPUT type="text" name="publisher">
    <INPUT type="hidden" name="bk" value="aa">
    <INPUT  type="submit"  name="submit" value="搜索">
</FORM>
```

图 2 网页表单的 HTML 代码

```
Type = select
Type = mulitpletext
Type = submit
Type = hidden
SelectName = type
TextName = title
TextName = author
TextName = publisher
HiddenName = bk
SubmitName = submit
SubmitValue = 搜索
HiddenValue = aa
FormName = search
Action = user
Action = getbook.asp
图书  类别  书名  作者  出版社
```

图 3 从网页表单中自动提取出来的特征

2.2 机器学习

目前有许多成熟的通过机器学习实现的文本自动分类方法，包括概率模型方法、关系学习方法、支持向量机方法等。其一般过程都是通过对已经分好类的一

组训练文本的学习来自动创建分类器,通过有指导的学习对测试文本进行分类。文中在对网页表单进行自动分类的过程中采取的是概率模型方法,并使用了朴素贝叶斯分类算法。

朴素贝叶斯分类算法是一种简单、有效而且在实际使用中很成功的分类算法,其性能可以与判定树与神经网络分类算法相媲美,在某些场合还优于其他分类算法。设有变量集 $U = \{A_1, \cdots, A_n, C\}$,其中 $A_1, \cdots, A_n$ 是实例的属性变量,C 是取 m 个值的类变量。假设所有的属性都条件独立于类变量 C,即每一个属性变量都以类变量作为唯一的父结点,就得到朴素贝叶斯分类器。使用朴素贝叶斯分类器进行分类的做法是:通过概率计算,从待分类实例的属性值 $a_1, \cdots, a_n$ 求出最可能的分类目标值。即计算各类 $c_j \in C$ 对于这组属性的条件概率 $P(c_j \mid a_1, \cdots, a_n)$,其中 $j = 1, 2, \cdots, m$,并输出条件概率最大的类标签作为目标值。应用贝叶斯定理可得:$P(c_j \mid a_1, \cdots, a_n) = P(a_1, \cdots, a_n \mid c_j)P(c_j)/P(a_1, \cdots, a_n)$,朴素贝叶斯分类算法假设条件是独立的。因此 $P(a_1, \cdots, a_n \mid c_j) = \prod_{k=1}^{n} P(a_k \mid c_j)$,同时由于 $P(a_1, \cdots, a_n)$ 对于所有类为常数,因此,只要计算 $\prod_{k=1}^{n} P(a_k \mid c_j)P(c_j)$ 就可以了。假设 $P(c_1 \mid M)$ 为待分类网页表单 M 是查询接口的概率,$P(c_2 \mid M)$ 为待分类网页表单 M 非查询接口的概率。只要 $P(c_1 \mid M) > P(c_2 \mid M)$,就可以判定网页表单 M 为查询接口,反之则为非查询接口。

3　实验与改进

为了验证文中提出的 Deep Web 查询接口判定方法的可行性,实验中笔者从爬虫程序随机抓取的网页中抽取了大量的网页表单,并进行手工分类,选取了 65 个 Deep Web 查询接口和 130 个非 Deep Web 查询接口组成的训练集,140 个 Deep Web 查询接口和 160 个非 Deep Web 查询接口组成的测试集,实验结果数据如表 1 所示。

实验显示,利用朴素贝叶斯分类算法可以对网页表单获得比较好的分类效果。为了进一步提升 Deep Web 查询接口判定的效率和准确性,还可以在自动分类过程中加入一些启发式的规则。例如有些网页表单有 TEXTAREA 控件和 PASSWORD 控件,根据实际经验可以直接判定这类网页表单不是 Deep Web 查询接口。另外可以为网页表单中的元素数量设置一个阈值,当一个网页表单中的元素数量低于这个阈值时,就可以将这个网页表单划分为非 Deep Web 查询接口一类。例如有些站内搜索的网页表单元素数量很少,仅有一个文本框和一个提交按钮,对这类网页表单无法获得足够的信息,因此可将它们划入非 Deep Web 查询接口一类。

表 1　实验结果数据

	预测查询接口数	预测非查询接口数
查询接口数	128	12
非查询接口数	20	140
查全率:91%		查准率:86%

4　结束语

Deep Web 查询接口的自动判定是获取 Deep Web 信息的基础,实验验证文中提出的判定方法具有良好的可行性,并取得了较好的判定效果。通过完善网页表单查询接口特征提取方法,可以使该算法对 Deep Web 查询接口的判定效果得到进一步提高。如何提高网页表单特征抽取质量将是下一步的主要研究内容。

参考文献:

[1] Ghanem T M, Aref W G. Databases Deepen the Web[J]. IEEE Computer, 2004, 73(1): 116 - 117.

[2] Bergman M K. The Deep Web: Surfacing Hidden Value[J/OL]. The Journal of Electronic Publishing, 2001, 7(1)[2001]. http://www.press.umich.edu/jep/07 - 01/bergman.html.

[3] Sherman C, Price G. The Invisible Web: Uncovering Information Sources Search Engines Can't See[M]. New York: Cyber Age Books, 2001.

[4] Bergman M K. Deep Web White Paper[EB/OL]. 2004. http://brightplanet.com/technology/deepweb.asp.

[5] Lage J P, da Silva A S, Golgher P B, et al.. Automatic generation of agents for collecting hidden Web pages for data extraction[J]. Data & Knowledge Engineering, 2004, 49: 177 - 196.