基于语义支持的 Deep Web 数据抽取

高 明 王继成 李江峰
(同济大学电子与信息工程学院 上海 201804)

摘 要 在分析 Deep Web 查询实现机制的基础上,给出了在语义本体的支持下,通过机器学习来实现自动填充查询接口,以实现自动数据抽取的算法:构造二维表,表的列为通过 Deep Web 查询接口页面提取到的各个控件,通过为各控件赋值的方式来为表中添加相应的元组,根据返回结果的情况,即数据抽取成功或抽取失败,作为指导进行分类学习,最终依照学习的结果来自动构造请求字符串完成数据的抽取。实验表明算法具有较好的效果。

关键词 数据抽取,语义,机器学习,深网
中图法分类号 TP391 **文献标识码** A

WDB Data Extraction Based on Semantic Support

GAO Ming WANG Ji-cheng LI Jiang-feng
(College of Electronics and Information Engineering, Tongji University, Shanghai 201804, China)

Abstract The paper presented an algorithm which fills the query interface by using machine learning based on the analysis of mechanism of Deep Web query. The algorithm is able to extract data automatically. Firstly, a 2D table is constructed. The columns of the table are controllers extracted from pages of the Deep Web query interface. Then values of the table are filled by giving values to all the controllers. Next, a learning of classification is going to be achieved according to the result whether the extraction of data successfully or not. Finally, the data is extracted by constructing request string automatically through the results of the learning. The experiment shows that the algorithm runs effectively.

Keywords Data extraction, Semantic, Machine learning, Deep Web

1 引言

随着万维网的发展,特别是 Web2.0 的出现,越来越多的网页使用即时生成的方式来产生。数据存储在后台数据库中,需要时根据用户提交的请求返回结果,然后根据模板格式化返回的结果即时生成相应的页面。

这些存储在数据库中的内容通常被称为 Deep Web,这种页面的一个显著特点是,它们不能由传统的搜索引擎通过静态链接直接得到[1],而是需要通过 HTML 表单提交查询,由服务器根据请求动态即时生成,并把结果返回给客户端。

广义上来说,Deep Web 的内容主要包含 4 个方面[2]:

1. 通过填写表单形成对后台数据库的查询而得到的动态页面;

2. 由于缺乏被指向的超链接而没有被搜索引擎索引到的页面,大约占整个比例的 21.3%;

3. 需要注册或者其他限制才能访问的内容;

4. Web 上可访问的非网页文件,比如图片文件、PDF 和 Word 文档等。

Deep Web 的数据存储在 Web 数据库(WDB)中,使用时,必须通过与之对应的 Web 应用程序,填写页面表单(Form),以请求-响应模式的交互式方式来获取其中的内容。由于当今的搜索引擎爬虫还不具备自动填充表单的能力,造成了绝大多数存储在 WDB 中的高质量资源因无法检索到而大量闲置。

由于每个 Deep Web 数据源都是一个完全自治的系统,从而给基于 Deep Web 的数据抽取带来了一定的困难。各国的研究人员针对 Deep Web 数据的自动抽取展开了广泛的研究。文献[4]提出了一种基于图模型的 Web 数据库采样方法,它可以通过查询接口从 Web 数据库中以增量的方式获取近似随机样本,并且利用已经保存在本地数据库中的数据生成下一次查询;文献[5]通过将抽取工作分为结果模式生成和数据抽取两个阶段,提出了一种基于结果模式的数据抽取机制;文献[6]设计和实现了一种新的针对 Deep Web 资源的搜索引擎系统,它能够获取 Deep Web 资源信息,并且利用这些信息抽取出结构化数据;文献[7]提出了一种基于视觉信息的数据记录抽取方法,该方法在一定程度上克服了现有方法对 HTML 源文件的依赖;文献[8]结合 DOM 树结构和视觉信息来发现和分离数据记录,属于基于 DOM 树分析的数据抽取方法,该类方法可以利用 HTML 标签的层次结构来实现数据的准确抽取。

到稿日期:2009-10-15 返修日期:2010-01-08

高 明(1973-),男,博士生,主要研究领域为计算机网络信息处理、语义 Web、知识工程,E-mail:gmingtj@163.com;王继成(1957-),男,博士后,研究员,博士生导师,主要研究领域为数据挖掘、智能系统;李江峰(1983-),男,博士生,主要研究领域为计算机网络信息处理、知识工程。

本文从 Deep Web 的实现机制着手，分析了其实现方式和数据请求提交表单，并在语义本体的支持下，通过机器学习实现了一种表单自动填充算法，实验验证了其对 Web 数据库中的资源具有较好的获取能力，可以在较小的代价下获得较高质量的 Web 数据库中的信息。

2 问题分析与解决思路

Deep Web 的数据抽取，本质上是从相应的数据库中获得所需的数据，当今的 Deep Web 系统大都是基于 MVC 模式实现的，其基本过程如图 1 所示。

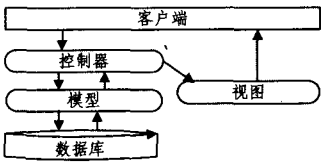

图 1 基于 MVC 模式的 Deep Web 数据抽取过程

1. 客户端提交查询请求给应用程序服务器；

2. 控制器负责提取客户端请求中的参数，并调用相应的业务逻辑模型，业务逻辑模型根据控制器传递过来的请求参数构造查询对象；

3. 业务逻辑模型调用相应的数据库连接模型。

数据库连接模型将构造好的查询由应用程序服务器根据客户的请求构造相应的结构化查询语句，然后将此结构化查询或查询对象提交给后台数据库服务器以获得相应的结果，应用程序服务器将此结果返回给相应的页面并转换成 HTML 代码返回给客户端。例如一个查询可由如下语句构造：

```
public Object doInHibernate(Session session) throws HibernateException{
    Criteria criteria = session.createCriteria(DocCondition.class);
    if(null != condition){
        //查询条件代码，例如
        if(null != condition.getTitle() && !"".equals(condition.getTitle()))
            criteria.add(Restrictions.eq("title",condition.getTitle()));
        if(null != condition.getCount())
            criteria.add(Restrictions.ge("count",condition.getCount()));
        //其他查询条件代码
    }
    return criteria.list();
}
```

而这个构造过程的关键就是通过客户端请求构造查询对象。

从代码分析可知，深网数据提取的关键是在数据库连接层查询的构造，而查询的构造的关键又是客户提交的数据，页面在向服务器提交查询时会将所有的输入内容生成一个查询字符串，在本例中就是：http://127.0.0.1/action/query.do?title=XXXX&count=XXXX&…，其中 XXXX 代表进行测试时输入的具体数据。由此，Deep Web 数据提取的问题可以从请求字符串构造的角度出发去考虑，即根据查询页面的结构构造一系列的键值对，调用相应的 action 来实现数据的自动提取。

本文采用的策略是，构造二维表，表的列为通过 Deep Web 查询接口页面提取到的各个属性，通过语义本体为各属性赋值的方式来为表中的列赋值，并提交请求。根据返回结果的情况（数据抽取成功或抽取失败）来对分类学习进行指导，最终依照学习的结果来自动构造请求字符串完成数据的抽取。

把二维表中的每一条数据看作是一个实体，各请求字符串的键值作为此实体的属性值来看待。通过实验得知，若某一属性值与能抽取到数据键值同种类型，且含义相近，则能从 Wdb 中获取到正确结果的可能性最大。

为此，以《同义词词林》[9]作为语义支持本体库来进行分析。《同义词词林》中定义了一种比较完善的汉语语义代码体系：以词义为主，兼顾词类，并充分注意题材集中。把词语分为大、中、小类 3 级，共 12 个大类，94 个种类，1428 个小类，小类以同义原则划分此群，每一词群以一标题词立目，共 3825 个标题词[9]。在此基础上，哈尔滨工业大学信息检索研究中心为适应信息时代的需要，推出了《哈工大信息检索研究室同义词词林扩展版》[10]，在原来 3 级分类体系的基础上增加了两级编码与原有编码集成，这样任一编码都可以唯一标识词库中出现的词语。

针对《同义词词林》中的分类编码做如下讨论：

令语义义类为词语集合 W 上的一个关系 R，则：

1. 对任一词语 $a(a\in R)$，它是与自身相关的，即存在 aRa，因此 R 是自反的；

2. 对于任意词语 $a,b(a,b\in W)$，若 aRb，即 a,b 属于同一义类，则必有 bRa，即 R 是对称的；

3. 对于任意词语 $a,b,c(a,b,c\in W)$，如若 aRb 且 bRc，即 a,b 属于同一义类且 b,c 属于同一义类，则必有 aRc，即 R 是对称的。

由此，通过定义一个表示 W 上语义的等价关系 R 来实现对 W 按词义进行分划，并分析 Deep Web 数据抽取的实现机制，同时考虑与能抽取到数据键值同种类型且含义相近的数据获取到可能性最大的实验结果，通过自动为二维表中属性匹配值来实现数据的自动抽取，匹配的值须同时满足与已知的能返回结果的匹配值同类型，且与前者同在一个语义分划内。

3 基于语义的 Deep Web 数据抽取

基于语义的 Deep Web 数据抽取分为两个部分，学习阶段和数据抽取阶段。由于无论是学习阶段还是数据抽取阶段都需要对表单中各个控件填充数据，且控件种类繁多，对此，将标签属性按类型进行分类，对不同类型的标签采用不同的方法进行处理。

3.1 控件分类

定义 1 查询接口 $I=\{i_1,i_2,\cdots,i_n\}$，$i_m=\langle p,T\rangle$，$m=1,\cdots,n$，p 为完成对查询结构构造的 form 中的各属性，$T=$ {text,textarea,file,password,hidden,radio,check,select,submit,reset,image,button} 为一枚举值。

定义 2 Deep Web 数据库检索模型：$WDDM=(I,p^+,D,res)$，其中 p^+ 为查询接口 I 中属性的集合。

这样就把对 Deep Web 的数据提取问题转换成了数据表的构造问题，下面所要做的就是为数据表的各个属性赋值并使之通过机器学习能够得到正确的返回值。

由于不同类型的属性处理的方式不同，根据 i_m 中属性 T 类型的不同来对其分类，情况如下：

1. text 可以匹配任何数据类型的属性，当其值被提交后，由服务器端根据需要进行类型转换，如果转换失败则将相应的异常返回给客户端。这是数量最多、最复杂的属性，是处理的重点，textarea 按照与 text 相同的方法进行处理。

2. radio 用于从多个值中选取某一个确定的值，由于此属性的值已经在页面中具体给出，因此以枚举的方式来处理这一属性；没有任何附加属性的 select 也以此方式进行处理。

3. check 用于对某一属性传递一个或多个值，以组合值的方式进行处理，主要是对其值进行合取；如果 select 添加了 multiple 属性，则也采用与 check 同样的方式处理。

4. hidden 属性主要用作为某一属性赋固定值，鉴于此，将其作为常量来进行处理，如果其他属性被标识为 readonly，则也按照此方式进行处理。

5. 可忽略的类型，包括：submit，image，reset 以及 button，这些属性的值并不在客户端提交请求时被提交到服务器端，因此在构造请求字符串时可以忽略这些值的存在；同时，由于 file 属性大多是在数据输入时使用，而在进行数据抽取时基本用不到此属性，因此也做忽略处理。

3.2 标签属性-值匹配

1. 对只有 text，textare，hidden 的表单的自动匹配

定义 3 text 查询映射 $m_{text}=(I,O,f)$，I 为查询接口，O 为与 I 对应的网络数据库类型相关的领域本体，f 为 O 中实体名或属性值到 i_m 键值的映射。

通过为 I 中各 p 属性从种子词库中随机抽取字符进行学习，并记录返回结果，如果服务器端返回状态码为 200，则表示学习成功，若状态码为 4xx 或者 500，则表示失败，由此构造的信息系统（二维表）如表 1 所列。

表 1 根据页面控件标签建立的信息系统

	属性 1	…	属性 m	结果
1	值[1][1]	…	值[1][m]	T/F
…		…		…
N	值[n][1]	…	值[n][m]	T/F

表中每一行代表一次关键词-值的匹配，其中值[n][m]代表第 n 个匹配中第 m 个属性的匹配值，结果一栏中 T 代表能返回正确的结果，F 代表不能返回正确的结果。

在进行学习之前，需要在本体库的支持下对属性的值做泛化预处理，以《哈工大信息检索研究室同义词词林扩展版》为例：

具体的泛化依据是对于名词，将其泛化为其所在词的词群，即泛化的结果为其第四级编码；

对于动词、形容词、副词泛化为其所在词的小类，即泛化的结果为其第三级编码。

2. 对 raido 标签属性的匹配

由于 radio 的值是固定的，如〈input type＝“radio” name＝“xxx” value＝“first” group＝“fenlei”〉，因此将 radio 的值看作枚举类型，匹配方式为：除 radio 之外的属性确定之后，以枚举的方式，在已确定的请求字符串后依次增加各个给出的值来构造多个完整的请求字符串。

3. 对 check 标签属性的处理

同 radio 类似，check 属性的取值也是固定的，二者差异之处仅在于 check 可同时取多个值。这也是唯一一个将数据以数组的方式提交给服务器的属性，因此一般采用将各个取值以排列的方式依次匹配，可能带来组合爆炸。由于 check 取值组合大多数情况下暗含一定的意义，本文对此属性暂时采用了先根据其含义手工组合，然后再进行数据匹配的方式来实现。

4. 对 hidden 的匹配

将 hidden 属性的值和注明了 readonly 的属性作为常量来处理，附加到请求字符串的后面发送。

3.3 算法实现

学习阶段算法基本思想：

1. 从种子库中随机抽取词汇，根据属性的类别，按照 3.2 节给出的策略为各属性匹配相应的值，并提交查询请求，同时将与关键词匹配的值存入信息系统；

2. 将服务器响应结果存入信息系统；

3. 对信息系统中的数据参照语义本体进行泛化，使用决策树进行分类，将能正确获取到服务器数据的情况存入模式库，供数据抽取参照使用。

算法描述：

```
begin proc
  construct_q _param();
  send_q _param();
  while(response&&! meet_end)
    if(resp_ok)
      generalizeProperty();
      insert_to_pattern_base();
    else
      discart();
    end_if
    construct_q _param();
    send_q _param();
  end_while
  classification_pattern_base_C4.5();
end_proc
```

数据抽取阶段与学习阶段类似，主要差异在于构造请求字符串时不再从种子库中提取数据，而是从语义本体中进行匹配。数据也不再存入库。

算法可以以其学习的方式来构造一系列的请求，以获取 Web 数据库中的数据，但是还存在着数据孤岛问题，主要原因为：种子词汇数量大小以及覆盖范围的局限性，导致返回结果大量局限于某一小范围。对此，可以采用的改进措施：(1)为了防止数据孤岛，从获取的结果中提取关键词扩充种子库；(2)某些模式可能获取不到数据，这时更改模式库和学习信息系统，必要时刻重新学习。

4 实验验证

使用搜狗全网数据库（SogouCA）精简版[11]中的两个文件构造了试验数据库，共 128 个文本文件，总共构造了约 64 万条数据。数据表以数据记录编号为自增主键。

实验环境：

Web 数据库管理系统使用 MySql 5.0.51b，Web 应用服务器为 Tomcat 5.5，Web 应用编程语言为 Java。

（下转第 174 页）

International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW06). 2006
[5] Ghawi R, Cullot N. Database-to-Ontology Mapping Generation for Semantic Interoperability[C]// Very Large DataBase '07. Vienna, Austria, 2007
[6] An Y. Discovering and Using Semantics for Database Schemas [D]. University of Toronto, 2007
[7] Barrasa J, Corcho O, Gómez-pérez A. R2O, an Extensible and Semantically Based Database-to-ontology Mapping Language [C]//2nd Workshop on Semantic Web and Databases. 2004
[8] Stojanovic L. Methods and Tools for Ontology Evolution[D]. University of Karlsruhe, 2004
[9] An Y, Hu X, Song I. Round-Trip Engineering for Maintaining Conceptual-Relational Mappings[C]//CAiSE'08. 2008
[10] Velegrakis Y, Miller R J, Popa L. Mapping Adaptation Under Evolving Schemas[C]// Proceedings of the 29th VLDB Conference. Berlin, Germany, 2003
[11] Yu C, Popa L. Semantic Adaptation of Schema Mappings when Schema Evolve[C]// Proceeding of the 31st VLDB Conference. 2005
[12] Almeida R B, Mozafari B, Cho J. On the evolution of wikipedia [C]// International Conference on Weblogs and Social Media (ICWSM'2007). Colorado, USA, 2007
[13] Noy N F, Klein M. Ontology Evolution: Not the Same as Schema Evolution[J]. Knowledge and Information System, 2004, 6(4): 428-440
[14] Sparql2sql[EB/OL]. http://jena.sourceforge.net/sparql2sql

(上接第 158 页)

客户端参考词库则提取了搜狗互联网词库(SogouW)[12]中被标识为 N(名词)、V(动词)、ADJ(形容词)和 ADV(副词)的任意一种类型,且词频在 1 亿 3 千万以上的 3014 条双字词汇。

试验从客户端参考词库中随机抽取词汇构造查询进行学习,以能正确返回记录的主键为成功标记。学习完成后开始自动提取数据。

碰撞处理:试验使用的是新闻数据库,由于数据库内容的原因,出现了一个请求可以返回多条数据库记录的情况,对于这种情况也按照返回一个正确结果计量,在匹配重复时按照主键最大的一条数据进行判断。

考虑到服务器端程序对结果可能产生的影响,针对在服务器端使用模糊匹配、精确匹配两种方式处理客户端提交数据的情形,分别进行了实验。

在服务器端使用模糊匹配时的结果如图 2 所示。

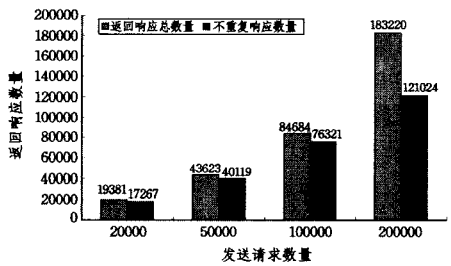

图 2 服务器端使用模糊匹配时算法返回结果数量

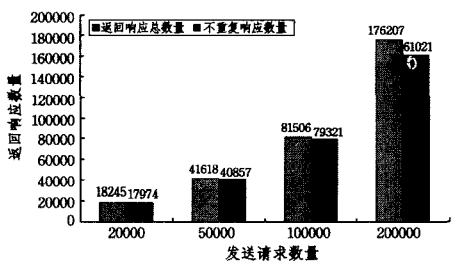

图 3 服务器端使用精确匹配时算法返回结果数量

可以看到,在客户端使用模糊匹配时,算法有较好的抽取效率,但是随着发送请求数量的增加,返回结果重复的情况也随之增加,这是因为模糊匹配可能导致多个请求被映射到同一条 Wdb 元组。

在服务器端使用精确匹配时的结果如图 3 所示。

当在服务器端使用精确匹配时,返回结果数量较使用模糊匹配时有所下降,但是算法的有效性增大,当客户端发起 20 万条抽取请求时,程序的有效性由使用模糊匹配的 60.5% 增加到了 80.5%。因此,该方式具有一定的实用价值。

结束语 随着 Deep Web 的迅速发展,针对 Deep Web 的数据抽取越来越成为一个研究的热点。每个 Deep Web 数据源都是一个完全自治的系统,因此给基于 Deep Web 的数据抽取带来了一定的困难。本文给出了一种用于 Deep Web 数据抽取的在本体支持下的机器学习算法,通过对 Deep Web 查询接口中的各个属性根据类型进行分类,进而采用不同的构造策略,在领域本体库的支持下可以自动地构造相应的请求字符串来完成 Deep Web 中数据的抽取。实验结果表明,本文提出的算法有较高的效率和准确率,可以以较小的代价获得较高质量的数据。

参 考 文 献

[1] Chang K C-C, He B, Li C, et al. Structured databases on the web: Observations and implications[J]. SIGMOD Record, 2004, 33(3): 61-67
[2] 刘伟,孟小峰,孟卫一. Deep Web 数据集成研究综述[J]. 计算机学报,2007,30(9):1475-1489
[3] Zheng Z, He B, Chan K C-C. Understanding Web query interfaces: Best-effort parsing with hidden syntax[C]// Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data. 2004: 107-118
[4] 刘伟,孟小峰,凌妍妍. 一种基于图模型的 Web 数据库采样方法[J]. 软件学报,2008,19(2):179-193
[5] 马安香,张斌,高克宁,等. 基于结果模式的 Deep Web 数据抽取[J]. 计算机研究与发展,2009,46(2):280-288
[6] 陈鹏,刘烈宏. 深度 Web 资源搜索关键技术[J]. 北京航空航天大学学报,2009,35(1)
[7] Liu W, Meng X, Meng W. Vision-based Web Data Records Extraction[C]// Proceedings of the 9th International Workshop in Web and Databases. New York: ACM, 2006: 20-25
[8] Zhai Y, Liu B. Web Data Extraction Based on Partial Tree Alignment[C]// Proceedings of the 14th international Conference on World Wide Web. New York: ACM, 2005: 76-85
[9] 梅家驹,竺一鸣,高蕴琦,等. 同义词词林[M]. 上海:上海辞书出版社,1983
[10] 哈尔滨工业大学信息检索研究中心[OL]. http://ir.hit.edu.cn/. 2009
[11] 搜狗全网数据库(SogouCA)精简版[OL]. http://www.sogou.com/labs/dl/ca.html 2007
[12] 搜狗互联网词库(SogouW)[OL]. http://www.sogou.com/labs/dl/w.html 2007