# 基于 XML 的自动学习 Web 信息抽取*)

冀高峰[1] 汤 庸[1] 道 炜[1,2] 吴桂宾[1] 黄 帆[1] 王 鹏[1]
(中山大学计算机科学系 广州 510275)[1] (广东天讯电信科技有限公司 广州 510620)[2]

**摘 要** 因特网给我们提供了巨大的信息量,在信息量极其丰富的 Web 资源中,蕴涵着大量有用的知识信息。信息爆炸而知识匮乏是当今人们所面临的一个很重要的问题。通过搜索引擎来查找信息将不容易定位到用户最感兴趣的数据上。而通过 Web 信息抽取的自动化实现,可以提高信息获得的效率。信息抽取可以从网络上分析和发现有用的信息,废弃冗余的数据,提取用户知识领域的知识。本文分析了基于 XML 的 Web 信息提取,讨论了相关技术在 Web 信息抽取中的应用并建立了相应的 Web 信息抽取模型,通过自动学习来获取信息抽取规则,实现 Web 信息的自动提取。

**关键词** 信息提取,半结构化,自动学习,规则库,XML

## Auto-learning Web Information Extraction Based on XML

JI Gao-Feng[1] TANG Yong[1] DAO Wei[1,2] WU Gui-Bin[1] HUANG Fan[1] WANG Peng[1]
(Department of Computer Science, Sun Yat-Sen University, Guangzhou 510275)[1] (Guangdong Tianxun Telecom Ltd, Guangzhou 510620)[2]

**Abstract** Internet provides us explosive information and involves massive important and useful knowledge within the abundant Web resources. Info explosion and knowledge deficiency are big troubles confronting modern civilization due to the inconvenience of locating the vital data interested by user via search engine. However, the auto-realization of Web info extraction could significantly enhance the efficiency of info.absorbing. It can also discover as well as analyze targeted info, discard redundant data and extract user-knowledge-domain-info. This article analyzes Web info extraction methodology based on XML, discusses related technology concerning application of such methodology, establishes Web info extraction model in order to realize auto-extraction of Web info via auto-learning the regulations of Web info extraction.

**Keywords** Info extraction, Semi structural, Auto learning, Regulation library, XML

## 1 引言

因特网对我们来说是一个巨大的信息源,信息爆炸而知识匮乏是当今人们面临的一个很重要的问题,因而从 Web 上提取用户相关领域的知识可以极大提高有用信息获取的效率。Web 信息获取主要有两种方法:通过搜索引擎查询或者进行 Web 信息抽取。前者主要是通过关键字匹配查询,根据用户的请求获取相应的文档,用户必须从获得的文档当中自己查找有用的信息。搜索引擎主要由网络爬虫(Web Scraper),索引数据库(Index Database)和查询服务(Inquiry Service)组成。网络爬虫在网络里以发现尽可能多的匹配网站为目标,查询服务则返回尽可能多的结果,这些文档并不考虑用户的知识领域,对用户来说并不容易定位到自己需要的资源上。Web 信息提取则从自动网络里分析和发现有用的信息,废弃并不需要的数据,可充分提取用户知识领域的知识。

本文的组织结构如下:在第 2 部分,讨论了相关工作。第 3 部分,建立了 Web 信息提取模型的框架,并对框架的结构进行探讨,包括了 http 请求的发送,html 文档到 xml 文档的转换以及抽取规则库的建立三个方面。最后对 Web 信息提取模型进行了总结。

## 2 相关工作研究

信息提取则是近二十年来发展起来的。信息提取在各个领域有着广泛的应用。主要的应用领域有:篇章分析技术、多语言文本处理、深层理解技术、时间信息处理、利用机器学习增强系统的可移植能力、Web 信息提取等等。其中很重要的一个方面是 Web 信息提取。Web 信息提取主要有两种方法,一种是知识学习方法,一种是机器学习方法。知识学习方法是针对特定的领域由人工编制抽取规则来提取信息,一般它的准确度较高,但对于要从大量格式不同的网页提取信息则需要耗费更多的时间和精力去编写抽取规则;机器学习方法则可以通过学习和总结规则,从而对付未见过的文本的信息的提取。但它要建立在大量训练数据的基础上。现在网络上大部分的网页是用 HTML 表示的,HTML 是一种半结构化的标记语言,半结构化文本没有严格的格式,比如电报的报文。在半结构化文本里存在着一些结构化的信息,我们可以利用其来进行信息提取。从 HTML3.2 到 HTML4.0 把大部分内容专注在了显示外观上的考虑,用来描述数据的表示、用户输入和控制的交互逻辑和外部多媒体对象[3]。这样一方面丰富了网页的各种效果,但数据与显示标记的混合也使得 HTML 的语法越来越混乱。标记的增多,让浏览器软件的开发商放宽了标记的撰写的规则。语法的宽松包容了更多的用户撰写时的错误,也破坏了整个语法的准确性。在 HTML 中虽然也包含着一些自由文本与结构化信息,但它总体上是半结构化的。文[1]对网页类型做了颇有用的定义:若能通过

*)基金项目:国家自然科学基金项目(60373081,60673135)、广东省自然科学基金项目(04105503,5003348)、教育部"新世纪优秀人才支持计划"资助项目。

识别分隔符或信息点顺序等固定的格式信息即可把“属性-值”正确抽取出来，那么，该网页是结构化的。半结构化的网页则可能包含缺失的属性，或一个属性有多个值，或一个属性有多个变体等例外的情况。若需要用语言学知识才能正确抽取属性，则该网页是非结构化的。

XML 允许我们为文本创建一个结构化的模型，它分离了数据结构与表示方法。W3C 在制定 XML 的设计目标中，其中一个是处理 XML 文档的程序应该容易编写。对于从 XML 文档中提取信息已经有很成熟的方法，所以对于非结构化的 HTML 中包含的信息，我们可以通过把 HTML 转换为 XHTML(XML 的子集)来进行处理。

## 3 基于 XML 的自动 Web 信息提取模型

图 1 是 Web 信息提取模型图，反应了一个总体的流程。

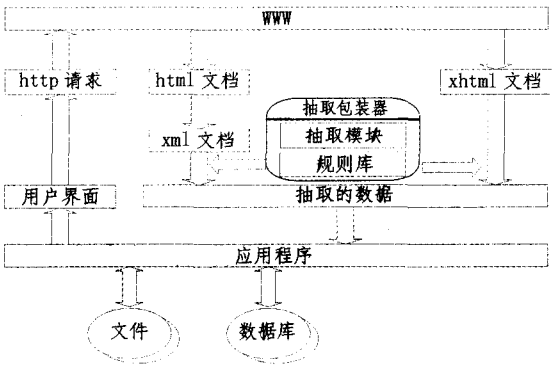


图 1 Web 信息提取模型

用户通过发送 HTTP 请求，获取相应的 HTML 页面(或者由程序自动请求页面，一般是请求大量页面的时候)，然后我们可以把 HTML 转换为 XML，然后通过一定的方法可以把 XML 里我们需要的数据提取出来(现在部分网站提供的页面是基于 XHTML 的，就可以免去 HTML 到 XML 转换这一步)。在程序中，我们可以把数据直接存到数据库，或者直接在页面显示给用户。用户可以根据显示结果选择不同操作，例如根据结果进一步发送 Web 请求，或者显示的数据就是用户需要的，也可以进一步存放到数据库。

以下对图 1 所示的框架图的流程进行相应描述。

### 3.1 发送 http 请求

当我们向网络上的服务器发送数据请求时，有的时候需要附带参数，比如用户名，密码，查询条件，或者表示选择了单选 box，复选 box，点击 button，则一般都是以表单的形式提交到相应的 URL。在 http 协议里，获取需要参数的页面一般有两种方法，一种是 get 方法，一种是 post 方法。Get 方法提交的数据量一般比较少，而且我们可以在浏览器的地址栏里看到附带在 url 的参数。Post 方法可以提交的数据量比 get 提交的数据量大很多。对通用情况我们可以采用 post 方法来做。

在这里，我们可以利用帮助分析提交数据的工具 URL Snooper。URL Snooper 是一个网络信息侦测(嗅探)软件，能够实时跟踪发送到网络或者从网络获取的数据信息。通过分析发送的数据包，可以详细了解提交到服务器的数据包以及从服务器发回的响应数据包的内容。

以下是一个 http 请求发送的数据包的具体例子：

```
POST /jwc/cgi/query_cj.asp HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
application/x-shockwave-flash, application/vnd.ms-excel,
application/vnd.ms-powerpoint, application/msword, */*
Referer: http://jwc.sysu.edu.cn/home/chengji/index.aspx
Accept-Language: zh-cn
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;
SV1; .NET CLR 1.1.4322)
Host: 202.116.64.13
Content-Length: 256
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: ASPSESSIONIDCCTTQTDA=EBBGBBAACIILGHDMBBJCMNGK
__VIEWSTATE=dDwtNzE3NDAxOTg403Q802w8aTwxPjs%2BO2w8dDw7bDxpPDU%2B
Oz47bDxOPHA803A8bDxvbmNsaWNrOz47bDxqYXZhc2NyaXBOOmNoZW5namkoKVw7
Oz4%2BPjs7Pjs%2BPjs%2BPjtsPGltQnRuRmluZDs%2BPs2e5Uofyy2BBmSXHhNC
tpYpANTD&xh=123456789&passwd=666666&imBtnFind.x=26&imBtnFind.y=9
```

其中，xh＝12345678 和 passwd＝666666 表示查找学号为 112345678 以及密码为 666666 的学生的成绩，其它参数代表的意义请参考 http 协议的规定。对于各种不同形式表单的提交，我们都可以对其提交的数据包进行分析并编写相应的代码来发送数据包。

### 3.2 HTML 到 XML 的转换

当服务器返回给客户端 HTML 页面的时候，我们可以把 HTML 页面转换为 XML 格式的文件。虽然我们可以直接从 HTML 文档中提取数据信息，但是现阶段访问 HTML 文档内容的方法并不灵活。加之存在 HTML 编码的不规范性，使得直接访问 HTML 文档内容相对麻烦。我们可以采取把 HTML 文档先转换为 XML 文档，而对于 XML 文档内容的提取则已经存在着很成熟的技术了(比如说 DOM 和 SAX)。

从 HTML 到 XML 的转换现在存在着很多的转换工具，特别是在开源代码方面有很多转换工具，并且这些工具在不断的完善之中，比如 Html Tidy 以及 Webharvest。

以 Webharvest 为例进行分析。

Webharvest 是用 java 编写的开源代码，它可以收集需要的 Web 页面，并从中提取用户需要的信息。

Webharvest 主要处理的对象是基于 html 或者 xml 的页面，通过 Webharvest，可以把 html 页面转变成格式化的 xhtml(xhtml 基本架构可以说是以 xml 为基础撰写的 html)。Webharvest 工作的原理大致如下：编写 Configuration 文件，(xml 格式的文件)，描述对相应的网页怎样转换以及怎样提取信息。在 java 代码里，会有相应的 processor 根据 Configuration 文件执行相应的过程。

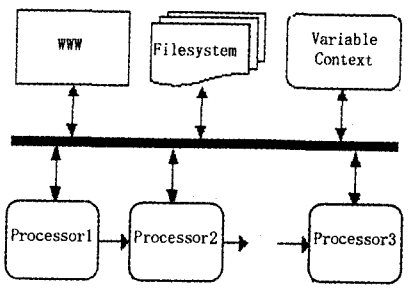


图 2 Webharvest 工作原理

以下是一个简单的配置文件的例子：

```
<?xml version="1.0" standalone="yes"?>
<config charset="UTF-8">
<file action="write" path="result.xml">
<html-to-xml>
<http method="get" url="http://someurl" charset="gb2312">
</http>
</html-to-xml>
</file>
</config>
```

该文件表明 Webharvest 获取 http://someurl 页面，并转换为 XML 文件，转换后的文件为 result. xml。其中，someurl 页面是 gb2312 编码的，Webharvest 在进行转换的过程中采用 UTF-8 编码。对于请求需要参数的 html 页面，则还需要发送其它参数。

假设上面的 XML 文件存储在 D 盘根目录下，文件名为 jiaowuchu. xml，则可以通过以下 JAVA 代码启动 Webharvest 执行配置文件定义的抽取过程。

```
Properties props = new Properties();
props.setProperty("log4j.rootLogger", "INFO, stdout");
props.setProperty("log4j.appender.stdout",
"org.apache.log4j.ConsoleAppender");
props.setProperty("log4j.appender.stdout.layout".
"org.apache.log4j.PatternLayout");
props.setProperty("log4j.appender.stdout.layout.ConversionPatter
n", "%-5p (%20F:%-3L) - %m\n");
PropertyConfigurator.configure(props);
ScraperConfiguration config = new ScraperConfiguration("d:/k/
jiao.xml");
Scraper scraper = new Scraper(config, "d:/k/");
scraper.setDebug(true);
scraper.execute();
```

通过以上代码，Webharvest 配置文件定义的抽取过程就得到了执行，用户只需在 Webharvest 配置文件中说明抽取的规则就可以抽取到不同的数据信息，包括 xml 文档，或者图片文件，音频文件或者其它的数据。

### 3.3 抽取规则库的建立

抽取信息的关键在于抽取规则库的建立。规则一般可以通过两种方法获得。一种是知识学习方法，一种是机器学习方法。知识学习方法是针对特定的领域由人工编制抽取规则来提取信息，一般它的准确度较高，但对于要从大量格式不同的网页提取信息则需要耗费更多的时间和精力去编写抽取规则；机器学习方法则可以通过学习和总结规则，从而对付未见过的文本的信息的提取。但它要建立在大量训练数据的基础上。下面建立抽取规则的自动学习模型。

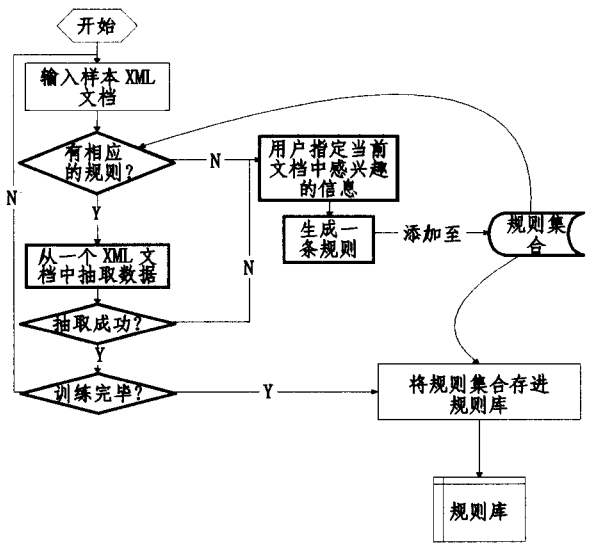


图 3 抽取规则自动学习模型

对于页面结构相似而并不完全一样的网页，可以通过少量的页面进行训练，建立抽取规则。然后，在建立起来的抽取规则的指导下，对大量结构相似的页面进行信息抽取，获得用户需要的信息。在上一步骤中，我们已经把 HTML 文档转成了 XML 文档。通过 DOM，我们可以把 XML 加载为一棵 DOM 树存放在内存中。对于一棵 DOM 树，某些叶子节点是用户需要的数据。在训练规则时，用户通过指定感兴趣的数据（DOM 树的某个叶子节点），可以得到 DOM 树中从根到指定叶子节点的一条路径（比如，用户指定需要查找 DOM 树中数值为“三星 E638”的叶子节点，这是很容易办到的，通过 DOM 规范中定义的方法即可）。这条路径就是一个规则。我们把这条规则存进一个规则集合中（初始为空）。对于不同的页面，结构是不尽相同的，所以用户感兴趣的某些同样的数据在不同的页面中，其抽取规则不同，我们则需要把这些不同的规则放在同一个规则集合中。以下算法描述了上述过程：

假设样本训练集为 samples_set，一个训练样本为 samples，规则集合为 rules_set，规则为 rules，用户指定感兴趣的知识信息的具体数据为 data。

算法描述：

```
rules_set=Φ;
while(samples_set!=Φ){
    samples=samples_set的某一项;
    if(根据rules_set成功抽取数据){
        samples_set= samples_set- samples;
        break;
    }
    rules=DOM树中从根到data叶子节点的路径;
    if(rules∈rules_set中){
        rules_set=rules_set∪{rules};
    }
    samples_set= samples_set- samples;
}
```

以上算法中，通过加大训练集以及当训练集里样本差别比较明显时，则会增加 rules_set 里规则的数量。规则数量的增多，使得系统具有从更多不同结构网页里提取数据的能力。

由多个规则集合所组成的规则库，将给抽取模块提供指导，由此可以抽取大量的结构相似的页面。以下以某个手机销售网站为例加以说明，从中抽取手机有关的信息。以下是某款手机信息在浏览器页面上的显示：

尺　寸：103.5×51×12.9mm
重　量：93g
萤　幕：240×320pixels、26 万色 2.12 时 TFT
相　机：300 万像素 CMOS
记忆卡：200 分钟
待　机：250 小时
电　池：800mAh
颜　色：極致黑、炫丽红
电磁波：SAR 实测值 0.578W/Kg

它的 HTML 代码如下：

```
<TD>尺　寸：103.5 x 51 x 12.9 mm</TD></TR>
<TR>
<TD>重　量：93 g</TD></TR>
<TR>
<TD>螢　幕：240 x 320 pixels、26 萬 色 2.12 吋 TFT</TD></TR>
<TR>
<TD>相　機：300 萬像素CMOS</TD></TR>
<TR>
<TD>記憶卡：microSD</TD></TR>
<TR>
<TD>通　話：200 分鐘</TD></TR>
<TR>
<TD>待　機：250 小時</TD></TR>
<TR>
<TD>電　池：800 mAh</TD></TR>
<TR>
<TD>顏　色：極致黑、炫麗紅</TD></TR>
<TR>
<TD>電磁波：SAR 實測值 0.578 W/Kg</TD></TR>
```

假设我们现在需要抽取手机“萤幕”这一项数据，在 HTML 转换为 XML 文档后，“萤幕”的抽取规则为：

rules=［〈HTML〉〈BODY〉〈TBODY〉〈TR〉〈TD〉〈TABLE〉〈TBODY〉〈TR〉〈TD〉〈TABLE〉〈TBODY〉〈TR〉［2］〈TD〉］

其中，rules 最后的［2］表示的是它是节点〈HTML〉〈BODY〉〈TBODY〉〈TR〉〈TD〉〈TABLE〉〈TBODY〉〈TR〉〈TD〉〈TABLE〉〈TBODY〉下面第 2 条〈TR〉〈TD〉的分支（第一条为关于“尺寸重量”的节点信息）。

以下是另一款手机具体参数在 HTML 页面的显示：

尺寸重量：94×50×20.8mm/115g

萤　幕：主 320×240pixels/1670 万色 2.2 时 TFT

副 128×160pixels/26 万 色 1.36 时 TFT

相　机：主 200 万像素 CMOS

副 33 万像素 CMOS

记忆插卡：microSD

使用时间：通话 210 分钟/待机 270 小时

电　池：950mAh

颜　色：黑、淡蓝

电磁波值：SAR 实测值 0.47W/Kg

它的 HTML 代码如下：

```
<TR>
<TD class=big11pt>尺寸重量：94 x 50 x 20.8 mm / 115 g</TD></TR>
<TR>
<TD class=big11pt>萤　幕：主 320 x 240 pixels、<FONT color=red><B>1670 万</B></FONT> 色 2.2 吋 TFT</TD></TR>
<TR>
<TD class=big11pt>　　副 128 x 160 pixels、26 万色 1.36 吋 TFT</TD></TR>
<TR>
<TD class=big11pt>相　机：主 200 万像素 CMOS</TD></TR>
<TR>
<TD class=big11pt>　　副 33 万像素 CMOS</TD></TR>
<TR>
<TD class=big11pt>记忆插卡：microSD</TD></TR>
<TR>
<TD class=big11pt>使用时间：通话 210 分钟 / 待机 270 小时</TD></TR></TD></TR>
<TR>
<TD class=big11pt>电　池：950 mAh</TD></TR>
<TR>
<TD class=big11pt>颜　色：黑、淡蓝</TD></TR>
<TR>
<TD class=big11pt>电磁波值：SAR 实测值 0.47 W/Kg</TD></TR>
```

相应地，“萤幕”参数的抽取规则表示为以下：

rules=［〈HTML〉〈BODY〉〈TBODY〉〈TR〉〈TD〉〈TABLE〉〈TBODY〉〈TR〉〈TD〉〈TABLE〉〈TBODY〉］［〈TR〉［2］〈TD〉＊］

其中，最后的＊号表示节点〈TD〉下面整棵子树的叶子节点的集合都为“萤幕”参数信息。当编写抽取程序模块时，就可以根据不同的规则对节点进行不同处理，这样就可以针对不同页面结构进行信息提取。此后，在这个规则集合指导下去抽取样本中别的课程的名称。如果成功，说明此条规则有效，若失败，则对新的样本中的信息也识别出它的路径（规则）。既然同样类型的数据存在不同的规则去抽取，则这些不同的规则都是抽取数据的规则之一，因此把规则集合不断扩大。

对于手机的颜色、电池、尺寸重量等信息，都可以类似地建立规则集合。当把所有的规则集合放在一起时，就形成了某类信息（手机参数）的抽取规则库。

**总结和下一步工作**　至此，信息抽取系统的几个关键技术已经完成，针对不同应用，可以把提取到的信息呈现给用户，或者存进数据库，或进一步根据信息发送 http 请求获得进一步的页面。针对不同网站，采取的技术需要相应地进行改变——对于获得的 HTML 中如若含有大量无用信息则可以在把 html 转为 xml 文件时就利用 Xpath 仅保留有用的信息，去除无用的信息；对于大数据量的 XML 文件可以采用 SAX 来提取信息以带来显著的性能提升；对于直接采取 XML 作为基础的页面（比如采用 RSS 技术的网站），则可以直接对其提取 XML 的数据。

网页抽取关键部分在于抽取规则的编写。抽取规则的获得是费时费力的。对于人工编写抽取规则来说，它的准确度相对较高，但工作量相当大，而且对于网页结构的动态变化，它的维护成本是比较高的。机器自动获取规则可以更好地适应网页结构变化的情况。

网页抽取是信息抽取的一个部分，网页一般属于半结构化的文本，除此之外，信息抽取也在结构化文本和自由文本抽取中有广泛的研究和应用。随着 XML 的普及，越来越多的网页将由 XML 来表现。XML 可以告知我们文档的意义而不只是形式。而抽取规则的构造方法的改进与升级，则是适应不断增长的动态网络的需要。

## 参 考 文 献


1　Line Eikvil：网上信息抽取技术纵览

2　胡东东，孟小峰．一种基于树结构的 Web 数据自动抽取方法

3　Birbeck M，et al. XML 高级编程

4　黄泳瑜，徐蕙英. XML 网页设计应用基础教程

5　张清军，朱才连．基于主动学习的 Web 页面信息抽取

6　张瑞，李石君．网上表格数据到 XML 的自动转换

7　尚福华，孙丽．基于 XML 的 Web 数据抽取方法的研究

8　谢维成，吕先竞，宋玉忠．基于 HTML 或 XML 描述的 Web 页信息抽取技术研究

9　张成洪，古晓洪，白延红．Web 数据抽取技术研究进展

10　Meyer E A. CSS 权威指南