第35卷 第18期 计 算 机 工 程 2009年9月
Vol.35 No.18 Computer Engineering September 2009
·博士论文· 文章编号：1000—3428(2009)18—0025—03 文献标识码：A 中图分类号：TP393

基于隐马尔可夫模型的Web信息抽取

刘亚清，陈 荣
(大连海事大学信息科学技术学院，大连 116026)

摘 要：针对Web信息抽取领域中存在的“项缺失”和“项无序”问题，提出一种基于隐马尔可夫模型的Web信息抽取方法。将Web文档解析为一棵扩展的DOM树，映射待抽取的信息项为状态，映射待抽取的信息项在扩展DOM树中的路径为词汇，使用归纳算法构造隐马尔可夫模型。实验结果证明该方法可以获得更好的抽取性能。

关键词：信息抽取；隐马尔可夫模型；扩展DOM树

Web Information Extraction Based on Hidden Markov Model

LIU Ya-qing, CHEN Rong
(Institute of Information Science and Technology, Dalian Maritime University, Dalian 116026)

【Abstract】 To solve disorder among information items and lack of information item in the field of information extraction, this paper proposes a Web information extraction algorithm based on Hidden Markov Model(HMM). It parses a Web document into an extended DOM tree, and maps an information item to a state with mapping a path in extended DOM tree about an information item to a vocable. An HMM model is obtained by using induction algorithm. Experiments show that the algorithm has better extraction performance.

【Key words】 information extraction; Hidden Markov Model(HMM); extended DOM tree

1 概述

当前，搜索引擎已经成为从万维网获取信息的主要工具之一。但是使用搜索引擎进行信息检索的结果往往包含大量不相关的Web页面，用户需要浏览每张结果页面才能获取所需的信息。解决这一问题的主要方法是开发相应的信息抽取工具。

目前，针对Web信息的抽取工具大致可以分为2类：基于特别语法的信息抽取工具和基于归纳学习的信息抽取工具[1]。

基于特别语法的信息抽取工具主要有TSIMMIS[2]。用户在使用这类信息抽取工具时首先需要将查询需求用特定的查询语言表示，然后系统根据自定义的语言解释器对Web信息源执行特定的查询操作。基于特别语法的信息抽取工具在进行信息抽取时一般依赖于信息的存储路径，并且掌握特定的语法对用户来说也是个沉重的负担。

基于归纳学习的信息抽取工具主要用于自动生成针对结构相似的Web页面的包装器。RoadRunner[3]是针对Web页面的结构进行归纳而形成抽取规则；STALKER[4]则是通过泛化抽取对象的上下文信息(包括HTML语法标记、空格、标点符号、单词)构建抽取规则。这类信息抽取方法没有考虑信息项之间的次序关系和信息项在Web页面中的位置特征，信息抽取模型的表达能力不足，从而影响了信息抽取的性能。

作为一种数学概率模型，隐马尔可夫模型(Hidden Markov Model, HMM)由于其强大的表达能力已经成功应用在非结构化的文本信息抽取中。HMM应用在Web信息抽取中的主要局限是：非结构化文本项之间的逻辑关系一般是“线性”的，而Web信息项之间的逻辑关系则是“非线性的”；传统的基于HMM的信息抽取方法是以目标项的上下文item作为词汇，而对Web信息进行抽取时，则会包含目标项的格式、位置等特征信息。

根据Web信息的特点，本文提出了一种基于HMM的Web信息抽取方法，给出了根据扩展的DOM树生成隐马尔可夫信息抽取模型的算法及相应的信息抽取算法。

2 隐马尔可夫模型简介

起源于生物学序列分析的隐马尔可夫模型是一种基于训练数据概率的多序列对比算法。形式上，隐马尔可夫模型是一个五元组 $\lambda=(S, V, A, B, \pi)$：

S 是状态集，$S=\{S_1, S_2, \cdots, S_N\}$；

V 是词汇集，$V=\{V_1, V_2, \cdots, V_M\}$；

$\boldsymbol{A}$ 是 N 阶的状态转移矩阵，$\boldsymbol{A}=\{a_{ij}=P(q_{t+1}=S_j|q_t=S_i),\ 1\leq i,\ j\leq N\}$，$q_t$ 表示 t 时刻 Markov 链所处的状态，a_{ij} 表示从状态 S_i 转换到 S_j 的概率；

$\boldsymbol{B}$ 是 $N\times M$ 观测值概率矩阵，$\boldsymbol{B}=\{b_{jk}=P(o_t=V_k|q_t=S_j),1\leq j\leq N,\ 1\leq k\leq M\}$，$o_t$ 表示在 t 时刻的观测值，b_{jk} 表示在状态 S_j 时 V_k 出现的概率；

π 是初始状态概率集合，$\pi=\{\pi_i=P(q_t=S_i),\ 1\leq i\leq N\}$。

3 信息抽取模型构造算法及信息抽取算法

信息抽取模型的构造过程如图1所示。首先使用HTML解析器将输入的Web文档解析为一棵扩展的DOM树，然后对待抽取的信息项在DOM树中的路径进行标注，最后使用隐马尔可夫模型中相应的公式对标注的结果进行归纳，获得信息抽取模型。

基金项目：国家自然科学基金资助项目(60775028)；大连市科技局基金资助重大项目(2007A14GX042)

作者简介：刘亚清(1979－)，男，博士研究生，主研方向：信息抽取，本体论；陈 荣，教授、博士生导师

收稿日期：2009-02-22 **E-mail：**liuyaqing234@yeah.net

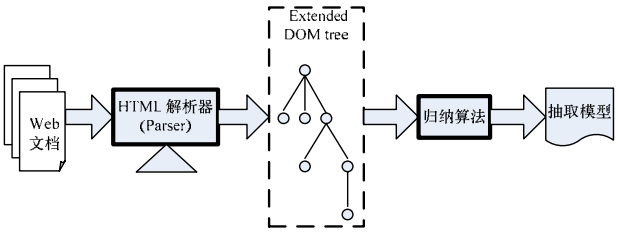


图 1　信息抽取模型的构造过程

3.1　扩展 DOM 树的生成算法

图 2 所示的文档是一个公司的相关信息，包括公司名、地址(包括街道、城市、州)、电话、传真和 E-mail。图 3 示出了图 2 所示文档对应的 HTML 标记序列，标记序列中可能出现的 term 分为 D, B, P 和 T 4 种类型。

IBM Corporation
4219 Southeast Admiral Court, Bartlesville, OK 74006

(918) 335-1775
(918) 337-6550 (fax)
mmerrifield@vnet.ibm.com

图 2　IBM 公司的 Web 文档

```
<SPAN><font color=blue size=-1>IBM Corporation</font></SPAN>
<SPAN><BR>4219 Southeast Admiral Court, Bartlesville, OK 74006</SPAN>
<DIV></DIV>
<SPAN><B>(918) 335-1775 </B><BR><B>(918) 337-6550 (fax)</B></SPAN>
<SPAN><BR><A>mmerrifield@vnet.ibm.com</A></SPAN>
```

图 3　图 2 文档的 HTML 标记序列

定义 1 D 是一个二元组(d_s, d_e)，表示 HTML 标记语言中具有开始标记和结束标记的标记类型，d_s 表示开始标记，d_e 表示相应的结束标记。dom(D)表示所有属于 D 类型标记构成的集合，例如(<B>, </B>)∈dom(D)。

定义 2 B 表示 HTML 标记语言中的 bachelor 标记类型。dom(B)表示所有属于 B 类型的标记构成的集合，例如
∈dom(B)。

定义 3 P 表示标点符号类型，dom(P)表示由所有标点符号构成的集合。

定义 4 T 表示一个单词。

下面给出节点的定义。

定义 5 节点(node)是一个三元组(type, text, children)，其中，type 为标记类型；text 为节点的文本；children 为孩子节点的集合。

扩展 DOM 树的构造过程为：(1)构造节点 rootnode= (root, null, null)作为扩展 DOM 树的根(root 只是为了方便问题的描述而增加的一个虚拟标记类型，null 表示空值)，并记 term_sequence 为输入的 HTML 标记序列。(2)初始化一个全局变量 t=term_sequence.getFirstToken()。(3)依次调用下面的函数 Parser1(rootnode)和 Parser2(PrimaryDOM Tree)构造一棵扩展的 DOM 树：

```
Parser1(current_node){
s="";
while(t!=null)
 if(t is a kind of ds)
    n=new node();    n.type=t;
    current_node.children∪{n};
    current_node.text=Parser1(n);
 else if(t is a de)
    current_node.text=s; return s;
 else
    s=s+t;
 end
 t=term_sequence.getNextToken();
 // getNextToken()用于获取 tokensequence 下一个标记
end
}
Parser2 (PrimaryDOMTree){
//PrimaryDOMTree 是调用 Parser1(current_node)后得到的一个初
//步的 DOM 树;
for each leafnode in PrimaryDOMTree //leafnode 为叶子节点
 term_sequence= split(leafnode.text,dom(P)∪dom(B));
 //按照 P 和 B 分割 leafnode.text 并将结果存于 term_sequnce 中
 for each term in term_sequence
      if(term is a kind of T)
        n=new node();
        n.type=term_sequence.getPreviousTerm()+
                "-"+term_sequence.getNextTerm();
        n.text=term;
        leafnode.children∪{n};
      end
   end
 end
```

算法的解释如下：函数 Parser1(current_node)主要根据当前的 token 类型是否为 D 的子类型构造 PrimaryDOMTree：如果当前的 token 是 d_s 类型，则构造一个新节点 n，然后递归调用 Parser1(n)；如果当前的 token 是 d_e 类型，则将 s(s 是一个用于存储连续出现单词的临时变量)的值作为当前节点 text 属性的值；否则，将 token 同 s 进行连接；在得到 Primary DOMTree 后，调用函数 Parser2(PrimaryDOMTree)可以得到一个扩展的 DOM 树：对 PrimaryDOMTree 树中叶子节点的 text 属性值以 dom(P)∪dom(B)的元素为分隔符进行分隔。对分隔后得到的每一项，如果项的类型是 T，则构造一个新节点，将当前项作为节点 text 属性值，将当前项的前一项和后一项用“-”连接起来作为节点 type 属性的值。

作为实例，图 4 是解析图 3 所示标记序列得到的扩展 DOM 树。限于篇幅，图 4 中只给出了每个节点的 type 属性的值，并且对于 D 类型的属性，只给出其 d_s 部分的值(同时省略了对应的 d_e 标记和 d_s 标记本身含有的属性)。

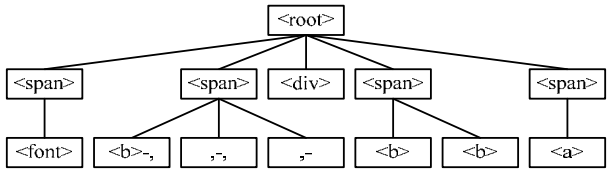


图 4　图 3 标记序列的扩展 DOM 树

3.2　信息抽取模型的确定

信息抽取模型的确定主要是解决 HMM 模型中的学习问题，即确定 HMM 模型的相关参数：对于已标记训练样本，一般采用 ML(Maximum Likelihood)算法学习参数，对于未标记训练样本，一般使用 Baum-Welch 算法学习参数。本文采用 ML 算法学习模型相关参数。

状态集 S 的确定通过用户在扩展的 DOM 树中对待抽取信息的标注确定状态集。例如对于图 2 所示的页面，待抽取信息有：公司名字(Na)，街道(Street)，城市(C)，州(State)，电话(T)，传真(F)和 E-mail(E)，则 S={Na, Street, C, State, T, F, E}。

初始状态概率集合 π 的确定：使用式(1)计算初始状态的

概率。

$$\pi_i = \frac{Init(i)}{\sum_{j=1}^{N} Init(j)}, 1 \le i \le N \quad (1)$$

其中，π_i 表示状态 S_i 的初始概率；$Init(i)$是所有训练序列中初始状态为 S_i 的序列个数；$N=|S|$。

状态转移矩阵 ***A*** 的确定：用式(2)计算状态转移的概率。

$$a_{ij} = \frac{C_{i,j}}{\sum_{k=1}^{N} C_{i,k}}, 1 \le i, j \le N \quad (2)$$

其中，a_{ij} 表示从状态 S_i 转移到 S_j 的概率；$C_{i,j}$ 是在整个训练集上状态 S_i 转移到状态 S_j 的次数；$N=|S|$。对于本文讨论的问题，其意义是指在 DOM 树中按照深度优先方法获得的状态序列。在图 3 所示的例子中，状态的序列为：Na→Street→C→State→T→F→E。

词汇集 V 的确定：每个标注节点的路径作为一个词汇，训练集上所有标注节点的路径构成词汇集。例如对于图 4 所示的 DOM 树，“公司名”节点所在的路径为“<root>→→<font color=blue sieze=-1>”，因此，状态 Na 的一个词汇为“<root>→→<font color=blue sieze=-1>”。

观测值概率矩阵 ***B*** 的确定：使用式(3)计算给定状态下某个观测值出现的概率。

$$b_{jk} = \frac{E_j(V_k)}{\sum_{t=1}^{M} E_j(V_t)}, 1 \le j \le N, 1 \le k \le M \quad (3)$$

其中，b_{jk} 表示在状态 S_j 时 V_k 出现的概率；$E_j(V_k)$表示在状态 S_j 下 V_k 出现的次数；$N=|S|$；$M=|V|$。

3.3 信息抽取算法

得到信息抽取模型后，按照如下步骤抽取 Web 信息：

(1)根据生成算法将输入的 Web 文档解析为一棵扩展的 DOM 树。

(2)按照深度优先方式提取扩展的 DOM 树中每一个 text 属性不为空的节点的路径作为观测值。

(3)采用 Viterbi 算法将待抽取的输入序列标记为最大概率的状态标签序列。

4 实验

4.1 实验数据

实验数据来源于 BigBook 网站，输入关键字 IBM 后查询得到 48 个网页，共 630 条记录。每条记录由公司名、街道、城市、州、电话、传真以及 E-mail 等项构成，某些记录存在项缺失的情况。在实验中，随机选择 100 条作为训练样例，然后在整个数据集上进行测试。获得的部分隐马尔可夫模型参数如下：

状态集：S={Na, Street, State, C, T, F, E}。

初始状态概率：在训练样例中，所有序列都以 Na 作为初始状态，因此，状态 Na 的初始概率为 1，其余状态的初始概率为 0。

状态转移概率：见图 5。

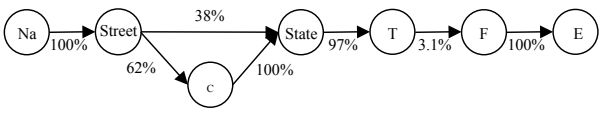


图 5 状态转移概率

限于篇幅，本文不再给出词汇集及观测值概率矩阵的实验数据。

4.2 实验结果及分析

为了便于对Web信息抽取的结果进行衡量和分析，采用召回率(recall)和准确率(precision)2个指标作为算法评价标准。实验结果如表1所示。此外，在相同的实验条件下和WHISK算法[5]作了比较，实验结果见表2。从表2中可以看出，本文方法进一步提高了信息抽取的准确率，而召回率高出不多的主要原因是个别出现概率极低的“词汇”没有被学习到。

表 1 实验结果

评价标准	Na	Street	C	State	T	F	E
召回率/(%)	95.31	94.23	90.11	93.23	94.25	92.18	91.24
准确率/(%)	100.00	100.00	96.36	98.77	98.04	99.68	100.00

表 2 算法比较

评价标准	WHISK 算法(5 次交叉验证)	本文算法
召回率/(%)	92.13	92.94
准确率/(%)	94.62	98.98

5 结束语

本文在对现有的信息抽取工具和方法进行深入分析的基础上，提出了一种基于 HMM 的信息抽取方法：首先将给定的 HTML 标记序列解析为一棵扩展的 DOM 树，接着在 DOM 树上对信息项及其所在的路径进行标注，然后使用隐马尔可夫模型中相应的概率公式计算模型所需的参数，从而获得信息抽取模型。实验表明该方法获得了更好的抽取性能。

下一步将主要从以下 2 个方面着手进一步提高信息抽取的质量：

(1)抽取粒度的进一步细化。考虑再一次使用机器学习算法解决某个被抽取的信息项包含 2 个或 2 个以上子信息项的问题，例如在图 2 中，“州名”和“地区代码”是作为一个信息项被抽取出来的。

(2)对于含有固定格式的信息项，考虑使用正则表达式作为约束来进一步提高信息抽取的准确率。

参考文献

[1] Laender A, Ribeiro-Neto B, Silva A, et el. A Brief Survey of Web Data Extraction Tools[J]. ACM SIGMOD Record, 2002, 31(2): 84-93.

[2] Hammer J, McHugh J, Garcia-Molina H. Semi-structured Data: The TSIMMIS Experience[C]//Proceedings of the 1st East-European Symposium on Advances in Databases and Information Systems. St. Petersburg, Russia: [s. n], 1997.

[3] Crescenzi V, MeccaG, MerialdoP. RoadRunner: Towards Automatic Data Extraction from Large Web Sites[C]//Proceedings of 27th Int'l Conference on Very Large Databases. San Francisco, USA: [s. n.], 2001.

[4] Muslea I, Minton S, Knoblock C. Hierarchical Wrapper Induction for Semi-structured Information Sources[J]. Autonomous Agents and Multi-Agent Systems, 2001, 4(1/2): 93-114.

[5] Soderland S. Learning Information Extraction Rules for Semi-structured and Free Text[J]. Machine Learning, 1999, 34(1-3): 233-272.

编辑 张正兴