基于网页分割的 Web 信息提取算法*

侯明燕,杨天奇

(暨南大学 计算机科学系,广东 广州 510632)

摘 要:针对网页非结构化信息抽取复杂度高的问题,提出了一种基于网页分割的 Web 信息提取算法。对网页噪音进行预处理,根据网页的文档对象模型树结构进行标签路径聚类,通过自动训练的阈值和网页分割算法快速判定网页的关键部分,根据数据块中的嵌套结构获取网页文本提取模板。对不同类型网站的实验结果表明,该算法运行速度快、准确度高。

关键词:网页分割;信息提取;聚类;阈值

中图分类号:TP311.5 **文献标识码**:A **文章编号**:1674-7720(2011)05-0054-03

Web information extraction algorithm based on Web page segmentation

Hou Mingyan, Yang Tianqi

(Department of Computer Science, Jinan University, Guangzhou 510632, China)

Abstract: This paper proposes a Web information extraction algorithm based on Web division to solve the high complexity problem of unstructured information extraction. The method adopts Web noise pretreatment, carries on the tag path clustering according to the document object model tree structure of Web. The key part of the Web is determined rapidly through automatic training threshold value and Web page segmentation algorithm, and Web text extracted templates are obtained according to nesting structure in the data block. Experimental results on different kinds of Web sites show that the algorithm is fast and accurate.

Key words: Web page segmentation; information extraction; clustering; threshold

信息抽取 IE(Information Extraction)是一种直接从自然语言文本中抽取事实信息,并以结构化的形式描述信息的过程。通常被抽取出的信息以结构化的形式存入数据库中,可进一步用于信息查询、文本深层挖掘、Web 数据分析、自动问题回答等。Web 页面所表达的主要信息通常隐藏在大量无关的结构和文字中,这使得对 Web 文档进行信息抽取十分困难。一般的网页内容包括两部分,一部分是网页的主题信息,如一张新闻网页的新闻标题、新闻正文、发布时间、新闻来源;另一部分是与主题无关的内容,如广告信息、导航条,也称为噪声信息。如何有效地消除网页噪声,提取有价值的主题信息已成为当前信息抽取领域的一个重要课题[1]。参考文献[2]提出一种依靠统计信息,从中文新闻类网页中抽取正文内容的方法,有一定实用性,但适用范围有限。参考文献[3]针对 Deep Web 信息抽取设计了一种新的模板检测方法,并利用检测出的模板自动从实例网页中抽取数据,但只能用于电子商务网站。参考文献[4]从网页中删除无关部分,通过逐步消除噪音寻找源网页的结构和内容,但提取结果不完整。

考虑以上方法的优缺点,本文首先对网页噪音进行预处理,通过自动训练的阈值和网页分割算法快速判定网页的关键部分,根据数据块中的嵌套结构获取网页文本抽取模板。

1 网页预处理及区域噪音处理

1.1 网页预处理

可以通过以下 3 个预处理规则来过滤网页中的不可见噪声和部分可见噪声:(1)仅删除标签;(2)删除标签及起始与结束标签包含的 HTML 文本;(3)对 HTML 标签进行修正和配对,删除源码中的乱码。

1.2 区域噪音的处理

为了实现网页的导航,显示用户阅读的相关信息,并帮助用户实现快速跳转到其他页面,网页中一般要设计列表信息,在处理此类信息时,本文设计了两个噪音

*基金项目:广东省软科学研究项目(2009B070300052)

识别参数。

Length=Length(content)为<tag>…</tag>标签内纯文本信息的长度，设定字符的ASCII code>255？length+2：length+1。

$$B_n=\frac{N_1}{N_2+N_1}\times\frac{NODE_{\text{nohref}}}{NODE_{\text{href}}+1}\times 100\% \tag{1}$$

其中，B_n为列表噪音判定系数；N_1是块中非链接字符的字数；N_2是块中链接字符的字数；$NODE_{\text{href}}$是块中有href属性的节点数；$NODE_{\text{nohref}}$是块中没有href属性的节点数。

2 基于启发式规则的网页分割

网页分割算法基于启发式规则，算法分为Xpath聚类和对聚类的Xpath进行分割两步。本文约定文档对象模型(DOM)树的叶节点按照其在原始HTML文件中出现的先后顺序编号。

(1)Xpath聚类。对具有最大相似度的叶节点进行聚类。节点取得最大相似度时，两个节点Xpath完全相同。本文用向量$X_i=\{x_{i,1},x_{i,2},\cdots x_{i,n}\}$表示第$i$个Xpath的聚类。其中，$x_{i,j}$表示第$i$个Xpath聚类中的第$j$个叶节点。定义节点间距为一个Xpath聚类中两个节点编号之间的间隔：

$$\Delta D_{i,j,k}=|x_{i,j}-x_{i,k}| \tag{2}$$

式(2)表示第i个Xpath聚类的第j个与第k个节点之间的编号间隔。定义平均周期为一个Xpath聚类中相邻节点间距的均值：

$$\Delta T_i=\frac{1}{n-1}\sum_{j=1}^{n-1}\Delta D_{i,j,j}+1 \tag{3}$$

定义间距方差为考察一个聚类中各个节点离散程度的量：

$$\sigma^2(\Delta T_i)_j=\frac{1}{n-1}\sum_{j=1}^{n-1}(\Delta D_{i,j,j+1}-\Delta T_i)^2 \tag{4}$$

(2)分割点。一个聚类中的不连续点称为分割点。为了反映分割点的具体位置，定义了一个变量θ，它是前后两个间隔之间的比值。

$$\theta=\frac{\Delta D_{i,(j+2),(j-1)}}{\Delta D_{i,(j+1),j}}=\frac{x_{i,(j+2)},x_{i,(j+1)}}{x_{i,(j+1)},x_{i,j}} \tag{5}$$

为了增强分割鲁棒性，为θ设定一个阈值范围。实验表明当$\theta\in[0.85,2]$时，可以得到较好的分割效果。算法采用如下启发式规则：(1)如果$\theta\notin[0.85,2]$，则将向量X_i在分割点处分割开。(2)如果一个向量的平均周期$\Delta T>$PreD，且没有进行分割，节点数目大于预定义值，则认为已经到达网页内嵌块聚类的边界。

3 算法描述

3.1 Xpath聚类算法

将一个目标页面表示为DOM树结构，采用深度优先遍历策略，提取DOM树中的每个叶节点。对于每次遍历的叶节点，通过比较其Xpath，将其序号添加到具有最大相似度的Xpath聚类中。具体算法描述如下：

```
Input  DOMTree
Output  XpathCluster
Cluster(DOM  Tree)
{  XpathCluster  =ϕ;
for  each  xpath  of  leaf  node
{
if  (XpathCluster.xpath.Find(xpath))
{XpathCluster.xpath.Insert(node);}
else
{XpathCluster.Insert(xpath);
XpathCluster.xpath.Insert(node);
}
}
Return  XpathCluster;
}
```

由于在聚类过程中，可能将非正文信息聚类到正文信息类中，因此先分析其方差。若一个聚类中的方差很大，则利用式(5)定位到分割点，将目标正文信息块与其周围的分隔噪音块分割开。另外，利用文本信息块的聚类平均周期、信息长度和HUB判别等统计参数，帮助定位分割信息条。当第1个满足全部启发式规则和统计信息的聚类出现时，可以认为已经找到了正文信息块，完成分割任务。分割算法描述如下：

```
Input  XpathCluster                                    //Xapth 聚类
Output  SegBoundary                                    //分割边界
Variables:Integer:Length_Threshold;
                                                       //正文长度的最小阈值
Float:B_n_Threshold;          //B_n 列表噪音判定系数的阈值
WebPageSeg
{  SegBoundary  =ø;
Count=0;
While(Count!  =XpathCluster.size())
{
If(XpathCluster.at(count).var0  is  within  threshold)
If(xpathCluster.at(count).size()>
                                  //MAXSIZE&&xpathCluster.at(cou
nt).length>  Length_Threshold
&&  xpathCluster.at(count).B_n>B_n_Threshold  &&  ΔT>
PreD  )  //check
{SegBoundary.insert(each  node  within  XpathCluster.at(count))
Break;
}
else  Count++;
}
}else{                          //利用启发式规则(1)进行分割
Detect  segment  point  use(2.3.4)
Sort(new  cluser);
Count++;
}
```

```
}
Return SegBoundary;
}
```

3.2 节点集合内的文本抽取算法

节点集合内的文本抽取算法描述如下：

```
Input SegBoundary[]; //分割出来的符合条件的文本块
Output TextHashMap<tagpath, table textchunk, document
        //frequency>基于 HashMap 的文本块模板映射
Variables Integer: Frequency_Threshold;
                    //table/div 嵌套次数的阈值
StringBuffer: textChunk;                    //文本块
For each chunkp in SegBoundary[]
While p has more HTML nodes
nNode=p.nextnode;
ifnNode is not table/div Tag
textChunk=textChunk+extracted text from nNode;
                    //抽取 nNode 间的文本信息
else if nNode is table/div Tag
{
if TextHashMap.contains(tagpath)==true
{ documentfrequency++;}
else{
Documentfrequency=1;
}
TextHashMap.put(tagpath, textChunk, documentfrequency);
}
While TextHashMap has more{tagpath, textChunk, document
                    //frequency}

h is TextHashMap's item
if document frequency of h≥Frequency_Threshold
Print textChunk of item h
```

3.3 阈值的确定

在上述算法中，需要设定 3 个阈值参数：*Length_Threshold*、B_n*_Threshold*、*Frequency_Threshold*，它们对算法的时间复杂度和抽取效果具有一定调节作用，处理网页结构相似的网页时，可以通过训练样本自适应地算出相应的阈值。对于不同类型网页的阈值，3 个参数的数据分布有较大不同，*Length*、B_n 的数据分布绝大多数处于较小范围内，这些数据也是需要去掉的噪音数据，因此，使用 K-means[4]对样本数据进行聚类处理，而 frequency 数据相对前两个参数没有明显的分布趋势，数据量不大，而且也处在{1-10}这样的一个较窄的局部区间中。实验表明，聚类分析效果不明显，因此本文用算数平均值求解。

(1)单个样本网页的阈值训练

$$Length\_Threshold=Mid(Kmeans(Length[X],Clusternum)) \quad (6)$$

$$Frequency\_Threshold=\frac{\sum_{i=0}^{Y-1} documentfrequency[i]}{Y} \quad (7)$$

$$B_n\_threshold=Mid(Kmeans(B_n[Z],Clusternum) \quad (8)$$

(2)M 个同类样本的阈值训练

$$Length\_Threshold=Min(Length\_Threshold[M]) \quad (9)$$

$$Frequency\_Threshold=Min(Frequency\_Threshold[M]) \quad (10)$$

$$B_n\_threshold=Min(B_n\_threshold[M]) \quad (11)$$

其中，$kmeans(Array[],Clusternum)$为聚类处理函数，$Array[]$ 为处理数据集合，*Clusternum* 为聚类数目，$Min(Array[])$为获取集合最小值。

本文设计一种新的文本抽取算法，该算法采用网页标签分割和 HTML 树结构，能获得较高准确度。整个算法简单实用，前期的去除网页噪音算法可以让抽取的网页正文信息更准确。在未来工作中，可以把该方法与现有中文信息处理技术相结合，如考虑文本信息的相关性以及文本的字体属性来判断其重要性。

参考文献

[1] 欧健文，霍守斌，蔡斌.模板化网页主题信息的提取方法[J].清华大学学报：自然科学版，2005，45(S1)：1743-1747.

[2] 孙承杰，关毅.基于统计的网页正文信息抽取方法的研究[J].中文信息学报，2004，18(5)：17-22.

[3] Yang Shaohua, Lin Hailue, Han Yanbo. Automatic data extraction from template-generated Web pages[J]. Journal of Software, 2008, 19(2): 209-223.

[4] GUPTA S, KAISER G, NEISTADT D, et al. DOM-based content extraction of HTML documents [C]. Proceedings of the 12th Word Wide Web Conference New York, USA: [s. n.], 2003.

[5] PELLEG D, BARAS D. K-means with large and noisy constraint sets [C]. Proceedings of the 18th European Conference on Machine Learning. Warsaw, Poland: [s. n.], 2007.

[6] 于琨，蔡智，糜仲春，等.基于路径学习的信息自动抽取方法[J].小型微型计算机系统，2003，24(12)：2147-2149.

[7] 周顺先.文本信息抽取模型及算法研究[D].长沙：湖南大学，2007.

(收稿日期：2010-11-02)

作者简介：

侯明燕，女，1986 年生，硕士研究生，主要研究方向：人工智能，数据挖掘。

杨天奇，男，1961 年生，教授，硕士研究生导师，主要研究方向：人工智能，神经网络，数据挖掘等。