WEB 文献资料采集系统①

马创新

(南京师范大学 文学院，南京 210097)

摘 要：为了能够充分利用 WEB 上丰富的文献资源，设计了一个专业的 WEB 文献资料采集系统 WLES。该系统集成了网页抓取和网页清洗两方面技术，并且引入机器学习方法到网页清洗中，通过机器对训练语料的学习得到一个清洗模型，然后用该模型来实施网页清洗。实验证明该系统在网页抓取和网页清洗方面都具有优良的性能，能够满足使用者的文献采集需求。

关键词：文献资料采集；机器学习；网页清洗；清洗模型

Web Literature Collection System

MA Chuang-Xin

(College of Liberal Arts, Nanjing Normal University, Nanjing 210097,China)

Abstract: In order to take advantage of the rich literature resources on the WEB, this paper designed a professional web literature collection system WLES. The WLES integrates Web crawling and Web cleaning technology. The machine learning method is introduced to the study of Web cleaning. Machine learning on the training data can get a clean model, and then use the model to implement web cleaning. Experiments show: WLES in web crawling and web page cleaning has an excellent performance, to meet the needs of the user's literature collection.

Key words: literature collection; machine learning; pages clean; cleaning model

1 引言

在信息技术迅猛发展的今天，WEB 上的文献资源十分丰富。为了能够充分地利用这些资源，我们设计了一个 WEB 文献资料采集系统 WLES（web literature collection system），给定一个起始网页地址，它能够自动获得这个页面上的所有链接；并且智能地分析这些链接所指向的页面内容，通过机器学习过滤掉页面中的杂质信息，采集到所需要的文献资料。

2 相关工作及问题描述

本系统集成了网络爬虫和文本清洗两方面技术，本系统网页抓取任务单一，仅需要采集网页上的文本信息，不必下载视图、音频等内容，也不需要深入多层去攫取网页，只要获取一个网页中所有链接到的内容。由于任务单一，所以就要求它有更高的采集精度，能够达到专业人士的实际使用需求。

从 WEB 中抓取的网页文件中会有很多与正文无关的杂质信息，如字体格式信息、广告、超链接等，需要对网页内容进行清洗才能得到所需的结果。前人所使用的方法主要是根据 html，xml 格式特点来构造分析树，而 Internet 上的网页并是根据某一种 WEB 开发语言来编写的，而且开发语言是在发展变化之中的，所以这类方法有缺陷。我们的研究思路与前人不同，不考虑具体的 WEB 开发语言，主要基于下述观察：

观察 1 任一部文献是由若干篇文档组成的集合，每篇文档保存在一个网页中。若文献 LIT 被划分成 k 个不重叠的带有文献内容的网页序列 DOC_i（$0< i \leqslant k$），那么 LIT={ DOC_1, DOC_2, …, DOC_k }，同一个序列中网页的开发语言和结构方式是大致相同的，甚至在在一个文献网站中的所有网页编写语言和结构方式

① 基金项目:国家社科基金重大项目(10&ZD117);江苏高校重点研究基地重大项目(2010JDXM023);江苏省教育厅高校哲学社会科学基金(2011SJB740010); 江苏省高校自然科学研究项目(11KJD520009)

收稿时间:2011-11-03;收到修改稿时间:2011-12-01

基本相同。

观察 2 任一网页是由若干个文本块组成的集合，即若网页文档 DOC 被划分为 m 个文本块序列 BLOj(0< j ≤m)，那么 DOC={BLO_1, BLO_2, …, BLO_m }。结合观察 1，LIT={ {BLO_1,BLO_2,…,BLO_{m1}}1, {BLO_1,BLO_2, … ,BLO_{m2}}2, … , {BLO_1, BLO_2, … , BLO_{mk}}$_k$ }。其中任意文本块是由 n 个字符 CHA 按顺序组成，即 BLO_j=<CHA_{j1}, CHA_{j2}, …, CHA_{jn}>。每个文本块都有一个结束标志。网页文档中的文本块可以分成两类，一类是负载广告、超链接等信息的杂质文本块；另一类是负载文献资料信息的正文文本块。

观察 3 任一个网页是由若干个段落组成的集合，即若网页文档 DOC 被划分为 x 个段落序列 PAR_y（0< y ≤x ），那么 DOC={ PAR_1, PAR_2, …, PAR_X }。任一段落是由多个文本块所组成，有些段落完全由杂质文本块所构成；有些段落完全由正文文本块所构成；有些段落中既包含杂质文本块，又包含正文文本块。网页中的杂质大多出现在正文开始之前和正文结束之后，如能判断出正文开始和正文结束的位置，就能够清洗掉大部分杂质信息。

3 WLES的总体设计

根据观察 1，构成一部文献的多个网页的结构方式基本相同，那么就可以任意选取其中一份网页作为训练语料 DOCi，并把其正文内容经由人工处理保存到另一文本中，作为机器学习的标准答案 ANSi，通过机器对 DOCi 和 ANSi 的学习，训练出清洗模型 MODEL，再用该模型对其余网页进行清洗，最终得到需要的文献资料。根据观察 2，任一个网页是由若干个文本块组成的集合，在训练清洗模型和对其余网页做数据清洗时都需要对文本做分块处理，而分块的依据就是文本块自身所带有的结束标记。机器学习到以下内容就可以实现网页清洗：1.“含有正文文本块的段落”开始的位置；2.“含有正文文本块的段落”结束的位置；3.“含有正文文本块的段落”内部需要删除的杂质。

图 1 展示了 WLES 的系统流程。该系统是由“网页抓取模块”、“机器学习模块”和 “网页清洗模块”三部分构成。首先用“网页抓取模块”从 WEB 上抓取相关网页；然后任取其中的一个网页 DOCi，并且把这个网页中的正文内容经由人工处理，保存在另一文本中作为标准答案 ANSi；以 DOCi 和 ANSi 作为训练语料，通过“机器学习模块”，得到清洗模型 MODEL；最后“网页清洗模块”利用该 MODEL 对抓取到的所有网页做数据清洗，得到用户需要的文献资料。

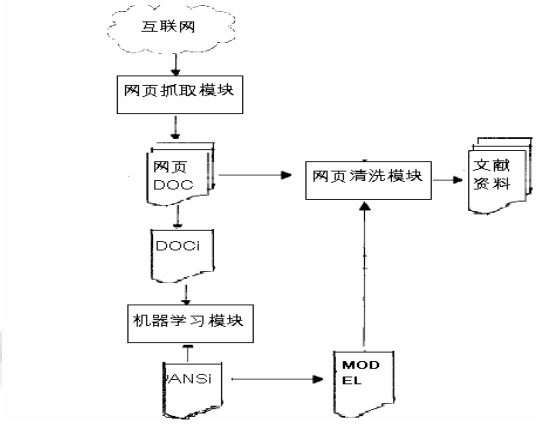

图 1　WLES 系统流程

4 WLES中关键模块和算法的设计

4.1 网页抓取模块

网页抓取模块主要是由“获取指定网页中的所有链接”和“保存所有链接的网页到文件”两部分组成。如果在 VC++的 MFC 环境下编程实现，要包含 afxinet.h、atlbase.h 和 mshtml.h 三个头文件，主要的代码如下：

```
//让浏览器控件打开网页；
CString sURL;
IECtrl.Navigate(sURL, ,NULL,NULL,NULL,NULL);
::OleInitialize(NULL);
//得到 HTML 文档对象；
LPDISPATCH pDocDispatch=m_IECtrl.GetDocument();
CComQIPtr<IHTMLDocument2,&IID_IHTMLDocument2> spDoc2;
spDoc2 = pDocDispatch;
//获取该文档中的所有链接；
CComPtr<IHTMLElementCollection> spElemColl;
HRESULT hr;
hr = spDoc2->get_links(&spElemColl);
……
```

读取某个链接网页中的数据以文本的形式保存到文件，可以用一个函数 CString saveHttpFile(const

CString& strURL)来实现，该函数的输入是链接网页的地址 strURL; 返回的是该网页数据的内容 strHtmlText。在主程序中可以探测到指定网页中的所有链接，然后通过循环调用这个函数，就可以把所有链接的网页以文本形式保存到磁盘中。

4.2 机器学习模块

4.2.1 文本分块

在机器学习和网页清洗时都要先对处理对象进行文本分块。我们定义了如下字符作为文本块结束标记 endMark：

.!?;,>}'), 。 ！ ？》 ； … 、” ’ ）

算法 1. 描述文本分块方案的函数 Text2Blocks(S)

输入：采用流形式读取的文件信息 S

```
Begin
1       for each element CHAi in  S
2          if endMark includes CHAi
3             if endMark donesn't include CHAi+1 then
4                insert "\n"between CHAi and CHAi+1
end
```

说明：如果在文件 S 中发现某个字符 CHAi 在结束标记 endMark 中存在，并且下一个字符 CHAi+1 在 endMark 中不存在，那么就在 CHAi 与 CHAi+1 之间插入一个换行符，从而实现文本分块。下面就是对某篇网页 DOC 做文本分块后的结果：

```
朱熹《论语集注》
</small>
</strong>
</p>
  </blockquote>
  <p align="left">
 
 
 
<font color="#000082" style="FONT-SIZE: 9pt">
此为书之首篇，
故所记多务本之意，
乃入道之门、
积德之基、
学者之先务也。
凡十六章。
</font>
<br>
……
```

4.2.2 作为训练语料的网页 DOCi 与答案文本 ANSi 之间的文本块对齐和段落对齐

对训练网页 DOCi 和标准答案 ANSi 分别进行文本分块之后，还要做 DOCi 与 ANSi 之间文本块的对齐，目的是把 ANSi 中的全部文本块与 DOCi 中所含正文内容的文本块对应起来，而 DOCi 中剩余的在 ANSi 找不到对应文本块的部分就是杂质信息。文本块对齐是通过计算文本块之间的相似度来实现的。

计算方法是：任给两个文本块 BLO_1 和 BLO_2，它们的所有字符构成的向量空间为 V={CHA_1，CHA_2，CHA_3，……，CHA_n}。文本块 BLO1 的向量 V_1 = { W_1，W_2，W_3，……，W_n }，其中 W_i 为 CHA_i 在 BLO_1 中出现的次数。文本块 BLO_2 的向量 V_2 = { $\psi 1, \psi 2, \psi 3$，……，}，其中 ψi 为 CHA_i 在 BLO_2 中出现的次数。计算这两个文本块相似度的公式是：

$$\text{Similarity(BLO1,BLO2)} = \overrightarrow{V1}, \overrightarrow{V2} = \frac{\sum_{i=1}^{n} Wi \cdot \psi i}{\sqrt{\sum_{i=1}^{n} \psi_i^{\,2} * \sum_{i=1}^{n} W_i^{\,2}}}$$

当两个文本块之间的相似度达到了规定的阈值 Threshold，就把这两个文本块对应起来。算法 2 描述了 DOCi 与 ANSi 之间的文本块对齐过程。

算法 2. 文本块对齐函数 AlignBlocks(ANS, DOC)

输入：分别作为文本块集合的网页 DOC 和答案文本 ANS

```
Begin
1 j=0
2   for n←0 to ANS.length()-1
3       for m←j to DOC.length()-1
4           if  Similarity(ANS[n],DOC[m])  ≥ Threshold  then
5              Alignment(ANS[n], DOC[m]);
6              j=m+1
7               break;
end
```

说明：第 4 行中，函数 Similarity(ANS[n], DOC[m])

是计算两个文本块的相似度，如果相似度达到规定的阈值就认为这两个文本块可以对齐。第 5 行中，函数 Alignment(ANS[n], DOC[m])的功能是实现 ANS[n]与 DOC[m]之间的对齐。

网页 DOCi 与答案文本 ANSi 之间的段落对齐就是把 ANSi 中的所有段落与 DOCi 中相应的既包含正文文本块又包含杂质文本块的段落对应起来。网页 DOCi 中杂质信息很多，并且段落的划分与 ANSi 也不一致。段落对齐是在文本块对齐的基础上实现的,目的是在网页 DOCi 中查找到作为段落开始的文本块和作为段落结束的文本块并做上标记。

我们设计函数 AlignParags(ANS, DOC)来实现段落对齐，函数的输入是分别作为文本块集合的网页 DOC 和答案文本 ANS。如果 ANS[n]是 ANS 中段落的第一个文本块，并且 DOC[m]与之对齐，那么就在 DOC[m]前面加上段落开始标志”BOP\t”；如果 ANS[n]是段落最后一个文本块，并且 DOC[m]与之对齐，那么就在 DOC[m]前面加上段落结束标志”EOP\t”。这样网页 DOC 就可以通过段落开始标志”BOP\t”与段落结束标志”EOP\t”分段，且每段中至少有一个正文文本块。

4.2.3 洗模型 MODEL 的构建

机器要学习到三个方面的信息，即：段落开始的位置标志、结束的位置标志以及段落中需要删除的杂质信息。在文本块对齐和段落对齐中，训练网页 DOCi 已经可以通过”BOP\t”与”EOP\t”进行分段，但这两类标志是添加在段落开始和结束的正文文本块前面的，还需要从正文文本块之前的杂质信息中提取出段落的开始和结束信息。我们选取三组数据，作为段落开始和结束的位置信息，分别存入六个向量中：1、所有段落开始之前的一个杂质文本块存入向量 Before1，结束之后一个杂质文本块存入向量 After1；2、所有段落开始之前的两个杂质文本块存入向量 Before2，结束之后两个杂质文本块存入向量 After2；3、 所有段落开始之前的三个杂质文本块存入向量 Before3， 结束之后三个杂质文本块存入向量 After3；

另外，机器还要学习到段落中需要删除的杂质文本块有哪些。实现了文本块对齐和段落对齐后，很显然，在网页 DOCi 各个段落中，正文文本块都与 ANSi 中的文本块实现了对齐，而那些无法对齐的文本块就可视为杂质文本块，被统一存入向量 vecImpurity 之中。

4.3 网页清洗模块

在这个模块中，我们用模型中的三组数据，分别对待清洗网页进行段落划分，然后经过机器自动比较，选用最优的一种段落划分方式，该方式的段落划分应该最为流畅并且得到的正文文本块的数量最多。当所有段落的开始位置和结束位置都被判断出来后，那么正文开始之前、结束之后以及段落之间的杂质信息都能被清洗掉。之后再利用向量 vecImpurity 中的信息清洗掉段落内部的杂质文本块，从而得到目标文本。因为在一个文献网站中所有网页的编写语言和结构方式基本相同，所以通过机器训练出一个模型后，该模型可对该网站所采集到的所有资料网页做清洗，而不需针对每部文献都分别训练出清洗模型来做网页清洗。

5 实验

本文用 VC++实现了该系统。为了验证其的性能，从文献资料网站 confucius2000、天涯在线书库、梦远书城等三个网站上选取了四种文献作为采集对象进行实验，从网页抓取和文本清洗两个方面进行评估。

在网页抓取方面，四种文献的七十八个网页全部被正确抓取。在网页清洗方面，用召回率和正确率两个指标进行考察。

Recall = 正确过滤掉的杂质文本块数 ÷杂质文本块数×100%

Precision =正确过滤掉的杂质文本块数 ÷WLES 过滤掉的文本块数×100%

实验显示网页清洗的 Recall 和 Precision 都达到 95%以上的占 92.33%，85%--95%之间占 4.82%，85%以下占 2.85%。

6 结语

针对 WEB 文献资料采集的特定需求，设计了一个 WEB 文献资料采集系统。该系统集成了网页抓取和网页清洗两方面技术，并把机器学习方法引入到网页清洗的研究中。本文用 VC++实现了这个系统，从实验结果来看 WLES 在网页抓取和网页清洗方面都具有优良的性能，能够满足使用者的文献采集需求。今后的工作包括：改进模型的数据结构；深入研究机器学习方法在网页清洗中的应用。

(下转第 37 页)

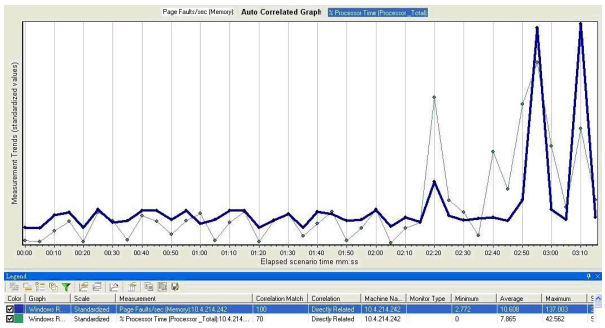

图 7 CPU 利用率与页面错误 Page Faults/sec 关联图

综上分析，发现“一般计划保存并继续”事务是系统性能瓶颈所在，需进一步优化修改。经程序代码排查，该段程序设计中，需要查询相关项目编号等信息，消耗较多的数据库资源。解决办法可考虑对 weblogic 应用服务器数据库连接池的配置优化，增大数据库的最大连接数。同时，从数据库设计和优化搜索算法两方面考虑修改此段程序，从而缩短响应时间，提高系统性能。

4 性能测试方法

在性能测试中，需要针对应用系统业务特点，模拟运行真实的业务场景，灵活运用 LoadRunner 性能测试工具，从性能指标和性能测试方法两方面分析入手，合理选择压力测试和负载测试等多种测试方法，运用多角度网页元素细分和关联度等分析工具进行详细分析测试性能指标结果，找出了该系统的性能瓶颈。性能测试中的测试分析是难点，在应用测试实践中，需要针对具体测试结果，调整测试分析思路，进而改进测试方法和步骤。同时，针对性能瓶颈，应该采用多种有效方法优化应用系统服务器参数配置和应用系统的程序代码，从而使整个系统性能提高，达到性能测试目的。

参考文献

1 陈霁,牛霜霞,龚永鑫.性能测试进阶指南.北京:电子工业出版社,2009.14–18.
2 桑圣洪,胡飞.性能测试工具LoadRunner的工作机理及关键技术研究.科学技术与工程,2007,(6):1019–1022.
3 黄文高,赵丹.LoadRunner 性能测试完全讲义.北京:中国水利邮电出版社,2010.6–8.
4 李艳芹,陈跃华,郭松柏.基于 Web 应用系统的性能测试综述.电脑知识与技术,2010,6(28):8014–8017.
5 董跃华,彭稷栋.利用 LoadRunner 实现网页负载压力测试.江西理工大学学报,2010,31(5):52–56.
6 Hewlett-Packard Development Company. What’s New in HP performance Center 11.00&HP LoadRunner 11.0,2011. http://h20195.www2.hp.com/V2/GetPDF.aspx/4AA1-2118ENW.pdf.
7 段念.软件性能测试过程祥解与案例剖析.北京:清华大学出版社,2006.10–12.

(上接第 12 页)

参考文献

1 Chen Ji H. Language specific issue and feature exploration in Chinese event extraction. Proc. of NAACL HLT 2009. Boulder, Colorado. 2009.209–212.
2 Metwally A, Agrawal D, Abbadi AE. Efficient computation of frequent and top-k elements in data streams. Proc. of the 10th International Conference on Database Theory (ICDT 2005). LNCS 3363, Berlin: Springer-Verlag. 2005. 398–412.
3 任仲晟,薛永生.基于页面标签的 Web 结构化数据抽取.计算机科学,2007,10:133–136.
4 刘书华,陈国奎.基于 PowerBuilder 的网页数据抓取.计算机系统应用,2009,18(2):171–175.
5 秦进,陈笑蓉.文本分类中的特征抽取,计算机应用,2003,23(2):45–46.
6 许建潮,胡明.中文 Web 文本的特征获取与分类.计算机工程,2005,31(8):24–25,39.