项目结题验收单

专家验收表（主持人所在单位组织 3-5 名专家对项目进行验收、自评。）

项目名称	基于泊松分布的科学数据监管平台的优化研究		
主持人	王寒冰	职务/职称	中级
所在单位	（加盖单位公章）东北农业大学图书馆		
专家意见	该课题研究的报申请书如实进行。研究成果具有较强现实意义，项目报告书内容详实，条理清晰，课题预期目标基本达成。报告撰写详实，文笔流畅，有针对性的选择使用参考文献，保证了课题研究的科学性与全面性。 经专家组全体成员审评后，一致同意可如期结题。 （如需要可增加页数）		
专家签字			
职务/职称	研究馆员	副研究馆员	副研究馆员	副研究馆员	副研究馆员

项目编号：2023010
注:项目编号请查看立项通知，也可缺省

CALIS 全国农学文献信息中心研究项目

结题报告

项目名称：基于泊松分布的科学数据监管平台的优化研究

项目关键词：科学数据；数据监管；平台建设；学科服务

项目单位(盖章)：东北农业大学图书馆

通信地址：哈尔滨市香坊区长江路 600 号

东北农业大学图书馆 306 室　150030

项目主持人：王寒冰

联系电话：18645022350

电子邮件：875307071@qq.com

提交日期：2024.5.10

基于泊松分布的科学数据监管平台的优化研究

王寒冰
东北农业大学　哈尔滨 150030

摘要：奶牛是畜牧业中重要的经济动物之一，原料乳的品质直接影响到各种食用 乳制品的生产。其中乳蛋白的含量是乳的主要评价指标，欧美奶业发达国家的 牛乳平均乳蛋白率可达 3.3%以上，以荷兰为最高，达 3.4%。而目前我国生鲜乳 的乳蛋白率标准已降至2.8%，普遍低于发达国家，原料乳品质差已经成为不争的事实。因此，在扩大饲养规模、增加奶牛存栏数量的同时，如何提高奶牛个体的乳产量和乳品质已经成为亟待解决的课题。

本课题以青年科学基金项目、承担“奶牛乳腺中 LAT1 基因对乳蛋白生物合成的调控研究”的东北农业大学乳品科学教育部重点实验室团队为服务对象。通过分析其科研过程中的学科信息服务实践可以发现，以科学数据管理服务平台为个性化学科服务模式，利用科学数据管理服务平台将散落在科研发现工作中的各种重要的学术实验数据进行收集、整理、分析、存储、加工以及再利用，发挥科学数据管理的作用，提高科学研究与创新的效率，为研究人员进一步的科研工作提供捷径，对于个性化学科服务的概念及图书馆建设和科学数据管理服务平台的建设具有深远意义，本课题在原有研究的基础上，对东北农业大学图书馆科学数据管理服务平台的建设进行优化研究，通过实证进一步说明了图书馆科学数据管理服务平台建设在提升图书馆服务层次与服务质量中具有重要作用。并结合实际情况，针对平台当前存在的问题，在以下两方面做出了优化研究：

(1) WEB 页面更新频率预测技术。该技术使用泊松分布来预测特定 WEB 页面更新变化情况，为爬虫的准备爬行提供基础。实验中使用的三个评价标准（准确率、命中率和覆盖率）来判别算法的性能。
(2) WEB 页面内容抽取技术。该技术利用 WEB 页面结构化标记 DOM 树以及 HTML Tag 的视觉线索来抽取 WEB 页面中最重要的内容。

关键词：科学数据；数据监管；平台建设；学科服务

1 课题研究的背景

新兴科学研究范式的兴起、数据驱动科研以及海量数据的整合与挖掘，为科技进步和科学研究带来了全新的发展视角。从 2018 年国务院发布的《科学数据管理办法》，到 2019 年国际科学理事会数据委员会（The Committee on Data for Science and Technology，CODATA）发布的《科研数据北京宣言》，再到 2021 年由中国科学院科学数据中心主任召开的 3 次联席会议，各项举措都表明了我国推动科学数据管理服务与开放共享的决心。明确了科学数据是独立的科研成果、重要的原创数据、科研的重要组成部分。科研人员在进行科学研究和科技创新活动中，面临科学数据的使用、管理、引用、共享及出版等一系列问题。高校图书馆作为服务于学校人才培养和科学研究的学术性机构，制定未来的发展战略需要与学校的“双一流”建设发展目标保持一致，需要为科研人员实施更精准有效的科学数据服务，也需要积极发挥自身数据资源优势，努力实现由信息中心向科学数据中心转型。近年来，我国高校图书馆科学数据研究注重对国外经验的吸收和利用、开拓了本土化研究领域。从服务方式、服务内容、合作机制等角度，调研美国、英国及加拿大等国家的大学图书馆科学数据管理服务现状，从调研结果角度总结了对我国科学数据管理实践的借鉴作用，对数据服务涉及的科学数据资源共建共享、馆员数据素养、用户科学数据素养需求、科学数据共享障碍因素等内容均有研究，指出“双一流”高校图书馆科学数据服务可以从数据素养教育、数据分析和数据挖掘 3 方面展开[1]。从科学数据服务、学科情报服务、智库建设服务 3 方面重新定位高校图书馆发展目标，助力于“双一流”建设之需。理论研究对促进高校图书馆科学数据服务的实践起到了积极的推动作用，为广泛了解我国科学数据服务水平及现状。

1.1 课题研究的目的和意义

科学数据是国家科技创新和发展的基础性战略资源，也是大数据时代最基本、最活跃的科技资源。高校是开展科学研究的重要阵地，是科学数据的重要产出者与利用者。随着数据驱动的第四科研范式的发展，高校图书馆依赖的知识创造、传播与利用环境正在从信息时代进入到数据时代，科学数据管理服务能力已经成为高校图书馆核心竞争力之一。构建一个成熟的科学数据管理服务平台帮助高校图书馆识别与评价其科学数据管理服务能力，进而实现服务能力的管理、改进和提高，具有重要的时代意义与需求。本课题在原有的相关信息服务平台基础上优化更加成熟准确的爬虫方式及网页信息抽取技术，构建高校图书馆科学数据管理服务能力成熟度模型，提供评价高校图书馆科学数据管理服务能力成熟度水平的框架，为高校图书馆提升科学数据管理服务能力明确发展的阶段和方向[2]。

1.2 国内外研究现状

大数据背景之下，数据规模海量增长，数据类型复杂多样，处理数据要求短时高效，科学数据日益重要。从 2018 年国务院发布的《科学数据管理办法》[3]，到 2019 年国际科学理事会数据委员会（The Committee on Data for Science and Technology，CODATA）发布的《科

研数据北京宣言》[4]，再到 2021 年由中国科学院科学数 据中心主任召开的 3 次联席会议[5]，各项举措都表明了我国推动科学数据管理服务与开放共享的决心。如何更好地提供科学数据服务同样是高校图书馆一直以来探索和努力的方向。

国内外许多学者针对高校图书馆科学数据服务展开了相应研究。Castle[6]详尽描述了将科学数据管理服务嵌入化学学科的亲身经验，同时指出了这种服务模式存在的不足；Berndsen[7]作为图书馆员设立了一门关于数据管理的培训指导课程，让学生在实践中理解科学数据管理。国内学者基于用户小数据[8]探讨了科学数据服务的内涵特点，构建了嵌入式科学数据服务体系模型；吴爱芝等[9]在明确科学数据服务及相关概念的基础上，阐述了高校图书馆科学数据服务的主要模式与内容；此外，王艳敏[10]针对性地研究了英国罗素大学集团成员图书馆的科学数据服务实践。国外学者 Inuwa 等[11]运用访谈法，从数据生命周期、数据共享、数据传播、数据素养教育等方面探讨了尼日利亚 3 所大学图书馆科学数据的情况；还有学者调查了约旦高校图书馆嵌入生命周期的科学数据管理服务现状，建议要充分宣传科学数据生命周期的改进服务[12]。国内学者徐菲[13]详尽分析了康奈尔大学图书馆的嵌入式科学数据服务模式；还有学者研究了基于科学数据生命周期理论构建高校科学数据管理服务模式[14]和科学数据管理服务内容体系的基本框架[15]。

2 主要研究内容及方法

2.1 主要研究内容

本项目以青年科学基金项目、承担“奶牛乳腺中 LAT1 基因对乳蛋白生物合成的调控研究”的东北农业大学乳品科学教育部重点实验室团队为服务对象。针对课题相关研究数据的网络页面实时下载问题、网络页面内容的精确抽取问题、文件信息的高效检索问题以 VIP 用户的身份设计了数据信息服务系统。其中爬虫模块保证了本项目系统对相关信息及时全面的监控，是本项目所描述系统中数据来源。网络页面正文抽取模块对网络爬虫下载的数据进行了清洗处理，保证了本项目检索结果的正确性。Lucence 搜索技术为我们的系统提供了高效信息检索技术。

文本将重点研究以下两方面的内容:

(1) 基于泊松分布的爬虫技术

由于本项目是基于东北农业大学乳品科学教育部重点实验室团队为服务对象，及时的信息收集和入库对提升用户满意度有重要意思。当本项目系统在新闻网站爬行页面时，必须时刻监控相关信息的更新,本项目结合了前人对网络页面更新频率的随机假设，提出了基于泊松分布的网络页面更新预测算法来指导网络爬虫高效的爬取页面。

(2) 网络页面正文抽取技术

网络页面是本项目在网上冲浪中浏览信息的基本单位。打开页面的源代码发现，一个网络页面中对相关信息的描述仅占整个页面 Code 的很少部分。本项目将研究 DOM 树结构以及隐藏在页面 HTML 中可视化信息，并根据这两种特征进行网络页面内容抽取。

2.2 关键技术优化

爬虫模块保证了本项目系统对相关信息及时全面的监控，网络页面正文抽取模块对网络爬虫下载的数据进行了清洗处理，保证了本项目检索结果的正确性。在本章节主要描述搜索平台所涉及中的两个关键技术，面向爬虫的网络页面更新频率预测技术，该技术使用泊松分布来预测特定网络页面更新变化情况，为爬虫的准备爬行提供了基础。在本章中本项目还实现了面向相关领域的网络页面内容抽取技术。该技术利用 WEB 页面结构化标记 DOM 树以及 HTML Tag 的视觉线索来抽取网络页面中最重要的内容。本项目中的实验证明，本项目实现的两个技术对可以有效地提高本项目系统搜索结果的准确性。

2.2.1 爬虫优化

网络爬虫本质上指的是一种综合性的适用于网络页面下载的应用程序，有效及高效的寻找并下载有价值的网络页面信息是设计者的终极目标，涉及的技术面较广[16]。伴随着现代飞速发展的网络技术和爆炸性增长的网络信息资源，传统的网络爬虫对于整个互联网来说早已经不能满足关于相关领域的信息检索工作，本章会就如何优化网络爬虫的信息抓取效率的问题进行研究。

2. 2. 1. 1 设计概述

一般来说，网络爬虫的具体运行流程可以这样描述：首先需要选取若干 URL（uniform resource locator）作为种子，然后在网络爬虫开始运行的阶段将种子 URL 放进爬行队列当中，然后某种优先级从爬行队列中按一定的顺序取出 URL，进行对应的链接资源下载，并针对下载到网络页面中的新的 URL 进行解析，将其放置在下载队列当中。不断的重复上述过程，直到 URL 队列为空或者 URL 队列满足其他的结束条件为止。用伪代码的模式描述了基本的爬行算法，

输入：URL 种子队列；

输出：指定的网站 WEB 页面内容信息

1. Web_Downloader(intial_urls) {
2. 将种子 URL 放入队列；
3. **While**(爬行队列不为空){
4. URL 从爬行队列中弹出；
5. 下载对应的 WEB 页面；
6. 抽取页面中新的 URL 以及相关的相关信息；
7. 将满足条件的 URL 放入队列；

从以上的代码当中可以看出来，对于网络爬虫来说，最关键的若干技术操作是 WEB 页面 URL 的解析、URL 的去重、抽取，URL 的过滤，以及 URL 的队列操作，还有对下载内容的保存等等。通常在网络爬虫的体系结构设计中将上面描述的每一个操作，全部设计成独立

的模块，因此，本系统中的统一网络爬虫体系结构的构成如下图 2-1 所示：

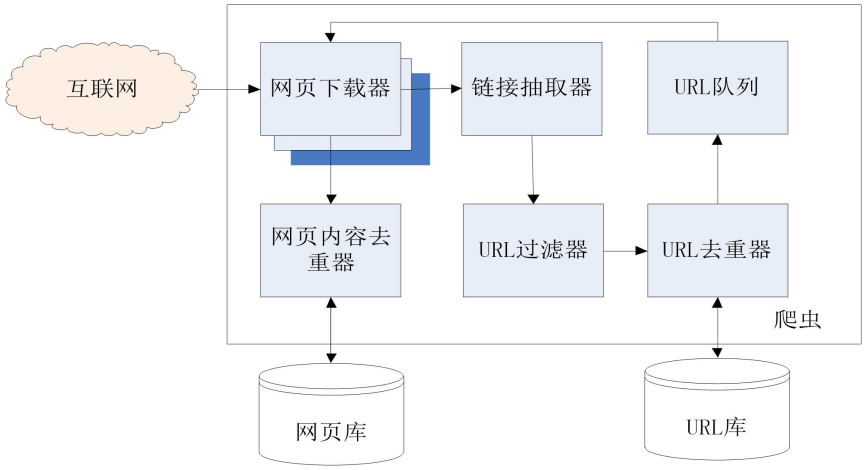

图 2-1 网络爬虫系统的主要工作组件

在信息服务平台的构建中， 本项目通过使用循环数组来实现 URL 队列，首先，循环数组要符合先进先出的网络爬虫下载规则，同时数组的内存空间要确定是一定的，在 URL 队列的内存空间不变化的同时 URL 保持增长；其次，平台实现了爬虫的断电回复机制，采用了在内外存缓存系统的基础上加定时恢复来实现。

2.2.2 专项爬虫

作为非常有针对性的学术领域，信息的采集和入库的及时且高效对于提升用户满意度有着重要的意义。当本项目构建的系统在网站页面爬行的同时，必须时刻保持监控相关信息的更新。譬如说系统每日需下载学术类网站，一般情况下该网站会将所有信息发布在一个信息列表页面上。网络爬虫的爬行需要随时监控这个列表页面的更新状态。当有新的相关的信息发布过来的时候，系统必须及时发现并下载到本项目构建的系统数据库中。

2.2.2.1 爬虫涉及的问题

对于相关领域中网络地址不变但是其中信息发生变化的情况，本项目系统将会使用优化爬行技术。然而，相关领域的优化爬行技术的实现存在着如下的挑战[17-19]。

（1）多次爬行：传统的下载器对每个独立的地址只下载一次，而爬虫会根据实际需求对每个 URL 爬行多次。那么面对大量的 URL，怎样选择恰当的重爬时间是本系统遇到的首个挑战。

（2）爬行的内容：面对千亿级别的网络页面，网络下载器的全网爬行行为成本过高。本项目构建的系统目的是在最短的时间内获取最新鲜的、及时的相关网络页面。那么，怎样有选择的下载并存储到的相关信息是系统遇到的第二个挑战。

（3）礼貌爬行：网络下载器应该做到可以准确的识别网络服务器所规定的下载规则，尽可能的降低对目标服务器载荷的压力。

2.2.2.2 优化爬虫性能的方法

关于爬虫的性能，我们之前的研究主要是以带宽的承载量和网络访问量去衡量，用网络爬虫的下载策略来提升其工作性能。本次研究将优化网络爬虫的下载模块，优化预测 WEB 页面的更新频率为下载的基础策略。

怎样判断页面中发布的信息是否已经更新，是本项目所设计的系统面临的第一个问题。用传统的办法来判断一个网络页面的内容是否已经更新，是比较两个时间节点上的 HTML 源码字符串的哈希值是否一致。数据集中常见的算法包括：位运算 Hash、加法 Hash、除法 Hash、乘法 Hash、混合 Hash、查表 Hash 等。大多数爬虫所采用 MD5 算法，因为 MD5 算法对字符串变化有较高的敏感性。而对体验度有较高要求的用户会选择计算方式简单、运算复杂度低的哈希算法。但是哈希法也有其明显的劣势，就是它对于信息内容字符串的变化非常敏感。在使用 MD5 算法的时候，哪怕只有一个字符不同，那么两个网络页面字符串的哈希结果的差距就会很大。在现实的网络中，往往会出现网络页面内容编码符号不同但是页面内容相似的情况。比如在网络页面 HTML code 中增加了一个空的“ ”标签，或者页面的中文本字体大小的变化，实际上，增加一些格式字符或者视觉标签，对于在网上冲浪的用户来说不能产生任何实质上的影响。再者，在页面中修改广告块或导航条，也不会对冲浪者的阅读体验产生干扰。所以说网络上的网络页面的内容更新具有非常明显的局部化特征，产生这种特征的主要原因是网络页面的半结构化，一般情况下的网络页面内容都是放置在页面的正中间位置，因此网络页面中间位置的文本变化是衡量网络页面内容是否变化的主要指标。将网络页面内容的变化转化成多维向量中各个分量值的变化。将图片个数、文本块大小、链接个数变化数量映射为多维向量。将网络页面的内容变化度投影到向量空间中，并使用 cosin 距离、欧式距离等方法针对网络页面向量之间的相似度进行判断。近来还有一些研究通过抽取网络页面中的中文本块来判断网络页面内容的重复性。Shingling 模式[20]将网络页面中的所有文本字符串都小写化，并且全都抽取出来，然后通过查找两个文档中的相同的 n 个字符序列的数量来衡量网络页面的重复性。但是在 Shingling 模式中，n 的大小决定了运行的效率，n 值太小则导致算法精准度不高，n 值太大则会导致算法的执行太慢。

我们主要研究的是爬虫如何重爬网络页面，优化网络爬虫在网络中的重爬策略。在以往研究中，大多数研究者采用三种策略，分别是基于信息生存周期的方法、基于网络页面变化的方法以及基于网络页面重要性的方法[21-25]。本项目优化的爬行技术是基于网络页面变化频率来研究[26-27]。假设页面内容的更新符合泊松分布定律，系统中的爬虫该什么时候去访问和下载网络页面，将通过对泊松分布参数的优化拟合来解决。对于一个统一资源定位符，页面内容的变化次数将使用 X(t)来表示从特定时刻 0 开始到 t 时刻来表示,λ表示页面变化的频率。同时假设所有的相关页面都是独立分布的，即所有相关页面都是符合泊松过程的特点。在一个区间，可以使用时刻 t 与 X(t)在时刻 t+1 的差的期望来表示相关领域网络页面发生变化的次数。在一般情况下，网络页面更新变化服从指数分布，在预测网络页面变化的频率之前本项目首先定义什么是网络页面的过时性与时新性。如果网络页面在前一个时刻与当前时刻的内容相似性在规定的访问之内，则称该网络页面是过时的。如果网络页面在当前时刻与前一个时刻的内容相似性在规定的访问之外，则网络页面是时新的。本项目定义某个样本网络页

面的时新性，那么在一个网络页面面的平均变化频率为λ的情况下，想要预测网络页面的更新时间，必须以网络页面更新频率λ进行准确估计为前提条件。在本项目中，由于实验条件的局限性，仅仅对相关网站进行了短期爬行实验，没有完整的、详实的、长期的记录样本网络页面的发布和更新的记录。所以只能在部分数据的泊松分布的参数上进行估计。在统计学中，对样本网络页面变化频率λ最简单的估计方法是在单位区间内网络页面样本更新的总次数 X 除以该时间段的长度之比[28]。由此，本项目得出某个相关网络页面的更新频率 r 的估计函数为：

$$\tilde{r} = \frac{\tilde{\lambda}}{f} = \frac{X}{n}$$

单位时间段内的页面更新的总次数用 n 来表示。上述的这种算法比较简单直观，在使用的过程当中会出现一些偏差。所以，我们采用 Crescenzi[29]的计算模型，来实现基于泊松分布的网络页面更新预测的算法。

输入：网络 B 页面更新的历史记录；

算法：

1. Predicting_Algorithm(Web 网络更新的历史记录) {
2. While(待预测 Web 网络页面的集合不为空){
3. 程序初始化；
4. For(每条等待预测的 Web 页面)
5. 计算 WEB 页面至今未更新的次数；
6. 计算参数 $\tilde{r}$；
7. 计算 Web 页面实际变化的概率；
8. 计算预测该页面变化的准确率；

为了体现在实现爬虫的过程中本项目算法的优越性，在相等的时间间隔内预测计算了网络页面更新频率的准确率，且与同类型计算方法进行对比研究，本项目在等量时间间隔时时监测相关网络页面的更新概率。

2.2.3 面向课题组的爬虫实验分析

本研究使用数据集来验证优化爬行策略的实际性能。基于课题相关数据的页面更新预测没有标准的数据集，所以本次实验会指定若干的相关网页以及公开数据库等，选取作为爬虫的初始链接种子对数据进行抓取。经过一段时间的爬行之后，系统在若干网页中收集 280000 多条资源定位符，随机选择若干条定位符进行追踪，记录网络页面的变化情况，并且根据页面变化进行算法分析。监测爬虫运行性能的指标为更新率、Divergence（差别度）、网页新鲜度与网页年龄[30-35]。本项目所构建系统的追求目标是所有网络页面保持最大的新鲜度和最小的年龄。通过对相关文献的分析，我们发现目前针对网页更新预测的评价标准并没有国际标准，在本文中将参考传统的数据挖掘研究中经常用来评价算法的查准率、查全率、

F-measure[36-37]这三个经常用于数据挖掘研究的传统评价算法。我们拟优化以上三个传统指标，其更加适合本研究的相关网页的更新预测算法。在实验中，将正确预测相关 WEB 页面的更新次数与数据集汇总所有网络页面变化的次数之比用 E_{right} 来表示，实际页面更新的总次数用 E_{total} 来表示。以此在实验中用来对应区间的平均准确度，这样可以直观的显示出本项目算法针对不同更新频度的网络页面预测的效果。将爬虫在下载页面的过程当中，监测到的实际更新次数用 $O_{updates}$ 来表示，将页面的下载总次数用 O_{visits} 来表示。将爬虫在下载中监测到的更新次数用 $O_{updates}$ 来表示，将 WEB 页面的实际更新次数为 $T_{updates}$ 。通过对本项目优化的预测算法进行性能衡量和分析，基于泊松分布的 WEB 页面更新预测算法准确率，命中率以及覆盖率都可以达到 91%以上，在综合类网站上，我们优化的预测算法较之前有所提高。

2.3 基于可视线索的 WEB 页面内容抽取技术

在前面的章节里，本项目解决了网络爬虫预测网络页面更新频率的挑战。本章将针对网络爬虫下载到页面所特有的一些性质，详细的设计并实现一种基于可视线索的网络页面内容抽取算法。该算法通过利用网络页面的可视化信息以及 DOM 树的信息，能有效地提取出一个网络页面的正文块。本章最后还对算法抽取效果进行了分析。为后续的搜索信息的实现抽取打下了坚实的基础。

2.3.1 WEB 页面视觉信息

网络页面是本项目在网上冲浪中浏览信息的基本单位[38-39]。当网络页面被浏览器打开的时候，各种各样的 WEB 页面元素（图片、文字、链接等）会在网络页面中以一定的格式呈现在本项目面前，其中包括板块布局，文字的大小，颜色，粗细以及图片和广告的位置。打开网络页面的源代码，本项目发现，网络页面中的 tag 是浏览器布局的基本单位。浏览器在呈现网络页面元素时，会把这些标签当成孤立的块，按照一定顺序放置到页面中。模块的顺序是根据网络页面 CSS 来指定的，在 CSS 中每个 tag 都标记自己的位置信息，如距离、宽度、边界等。CSS 还保存了 tag 元素的其他的样式信息，如颜色、字体、边框、背景等。

在本项目中本项目把上述的 tag 信息都称为网络页面的可视化信息(Visual Information)。表 2-1 是利用火狐浏览器的一个插件 firebug 得到的部分可视化信息。

表 2-1 可视化元素

信息名	值
font-family	字体类型
font-size	字号大小
font-weight	字符宽度
font-style	字体风格
color	颜色
line-height	线条长度
text-align	文本对齐
vertical-align	垂直对齐
width	页宽
height	页高
top	顶部对齐
right	右部对齐
bottom	底部对齐
left	左部对齐

这些信息被放置在网络页面面 Tag 元素的 class 中。很多传统算法都使用视觉线索，帮助程序自动识别网络页面的主要文本内容。

2.3.2 基于页面可视化线索的 WEB 页面内容抽取技术

得到上述视觉信息后，首要的任务如何使用 DOM 树结构以及这些可视化信息，以往的传统算法往往只使用 DOM 数结构或者只使用可视化信息[40-43]。本项目将结合网络页面的这两种特征，进行网络页面内容抽取。本项目拿到的最终抽取算法是若干条启发性规则，这些规则中包含了对网络页面 DOM 结构和视觉信息的使用。本项目提出的启发性规则都是通过对大量的相关网络页面进行分析最后统计出来的。所以，本项目中的算法在具有良好的泛化性。

2.3.2.1 Html 页面的载入过程

通过实验分析，本项目发现浏览器的网络页面生成过程就是，网络页面的各个标签渲染其 CSS 属性的过程，一个 HTML 页面加载和解析流程如下[44-47]]：

（1）用户指定 URL 后，浏览器使用该 URL 对网站服务器进行请求。请求成功后，服务器返回 URL 所指向的 HTML 源文件。

（2）浏览器接拿到所有的文件以后，开始按顺序加载 HTML 代码中的各个 tag。同时浏览器还会想服务器请求各个 tag 需要的其他资源，例如，某个＜head＞标签引用了外链式的 CSS 文件以及其他图片数据、JS 代码等。

（3）浏览器根据加载 CSS 样式渲染网络页面中的各个 tag，最后用户就可以看到色彩缤纷，

并且布局良好的页面。

从以上 3 个步骤中，浏览器显示网络页面的过程，就是对网络页面 tag 标签的放置和对标签的 CSS 格式解释、渲染的过程。网络页面 tag 块的所有布局、特征信息都放在 DOM 树结构和 CSS 视觉信息里面了。

2.3.2.2 Html 页面可视化信息的获取

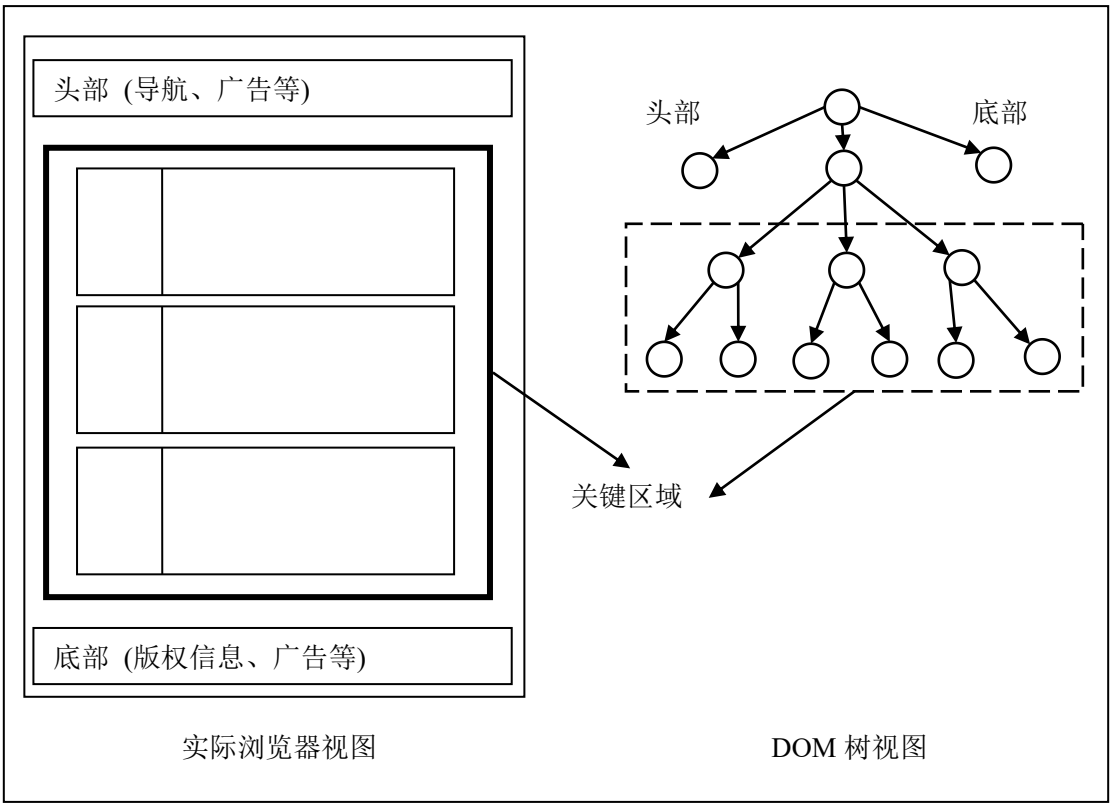

图 2-1 浏览器视图与 Dom 树视图的不同

图 2-1 是本项目队一个相关网络页面的抽象图，由图可以看出网络页面视图与网络页面 HTML 代码转化的 DOM 树视图对一一对应。网络页面正文的关键区域也和网络页面 DOM 树子树对应。

由上述分析本项目得知，网络页面的正文块既与网络页面视觉信息相关，又与网络页面 DOM 数结构相关，于是本项目产生下面的网络页面正文块特征规则。

（1）可视信息规则

宽度规则 （WR）：相关 WEB 页面中的主要内容块一般占浏览器视图的主要部分。本项目规定，网络页面的主要内容快的大小一般占浏览器宽度的一般以上，比如说普通浏览器是 960 像素宽，那么主要内容块的大小就占 480 以上。

高度规则（HR）：相关网络页面中的主要内容块的高度必须至少大于 60 像素。因为发布相关信息的内容一般都在 3 行以上。网络中文本行的高度最小是 20 像素，3 行就是 60 像素。

（2）Tag 规则

Tag 规则 （TR1）：相关网络页面中的主要文本信息块的 tag 节点的名字一般为 DIV

Tag 集合规则 （TR2）：网络页面主要内容 Tag 块的子树包含相似的 Tag 标签集。Tag 标签集指的是 Tag 标签名字的集合，tag 标签名字如<A>、<UL>、<SPAN>等，这里可以通过对比同一网站两棵 DOM 树的主要内容 Tag 节点中出现的所有不同的标签名字，。

（3）时间规则

由于相关信息剧透时效性。本算法将查找利用相关信息页面中的日期时间，将其视为一个分块的依据。

2.2.2.3 基于启发式规则的 WEB 页面正文算法抽取描述

基于启发式规则的网络页面正文抽取算法核心还是基于 DOM 子树 Tag 标签的视觉特征的。微软亚洲研究院的 VIPS 算法首先直接根据标签的视觉特征对网络页面的 DOM 树进行模块的划分，分完模块后。再根据子树模式进一步地去过滤不是正文部分的节点块。最后留下的模块就是网络页面正文部分。

在本项目中本项目首先读取一个 WEB 页面的 HTML 源文件，然后调用浏览器核心进行网络页面的渲染，从而得到一棵经过渲染的并且包含有可视化信息的 DOM 树。接着本项目的算法根据本项目定义的视觉规则，Tag 规则以及时间规则，查找找到网络页面的正文模块。本项目提出的基于启发式规则的相关正文模块查找算法过程见图 2-2。

算法：TextSearchNode
输入：
- 网络页面 html code；
- rc，最小子树阈值。

输出：正文节点 resultNode。
方法：
1 获取 web page 的代码，导入浏览器核心，找到节点文档 root；
2 根据视觉规则、tag 规则找到符合标准的节点集合；
3 根据时间规则，正文模块 resultNode；
4 返回 resultNode

图 2-2 基于子树可视化信息的页面主块查找算法

本项目的算法过程主要分 3 步：1）页面代码转换 Dom 树、启发式规则过滤、主模块标注。

（1）视觉信息树的生成

本项目网络爬虫在下载到一个 URL 的时候，一般只能下载到网络页面的 HTML 源代码。然而 HTML 源代码只包含网络页面的 Tag 信息。网络页面的视觉信息包含在网络页面外部的 CSS 文件中。在本项目中本项目不去下载和分析网络页面的 CSS 文件。本项目通过调用火狐浏览器的内核，构造一个虚拟浏览器。通过这个浏览器，本项目将原始的 HTML 文件，转换成为一个带有视觉信息的 DOM 树。 其中火狐浏览器的内核信息被称为 XPCOM，其 API 的用法在后面的代码部分可见，大题的转换过程见图 3-3。

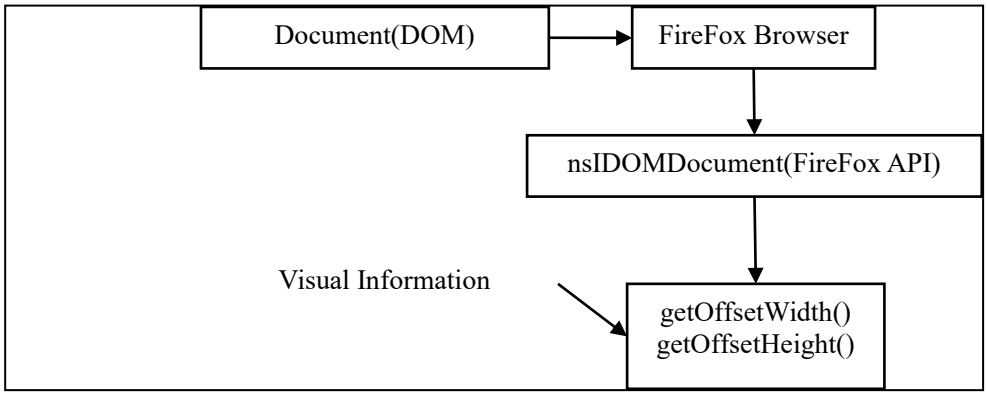

图 3-3 可视化信息的获取过程

（2）启发式规则过滤

本项目得到带视觉信息的 DOM 树以后，通过对树的遍历。过滤和检测每一个树节点，从而得到本项目想要的网络页面文本主模块。本项目的算法采用先根遍历法来遍历 DOM 树。在遍历过程中本项目认为网络页面的正文节点就隐藏在当前 DOM 树的各种 tag 标签内。本项目在遍历 DOM 树的时候首先访问树根结点然后访问当前节点的最左边子树，然后从左到右访问当前节点的其他子树。在遍历当前节点的子树时，仍然先访问根结点，然后从左至右遍历树的子树，当前节点已经没有子树则返回空。

输入：判断一个 DOM 节点是不是主要文本块。
输出：Ture 或者假。

```
isMinBlock(root)
    1  初始化队列 Queue，Queue 保存了 root 节点的所有孩子。
    2  While Queue 不为空
    3        E = Poll from Queue
    4        If E 符合视觉规则  and tag 规则  and  时间规则
    5              Then return ture
    6        Else  把 E 所有的孩子 pull 入 Queue
    7        End if
    8   End while
    9   Return false
```

图 3-4 主要文本块块的判断算法

实验显示本项目针对网络文章页面进行的主要内容块发现算法，划分出来的文本块一般是唯一的，不会找到两个以上的正文块。如果算法找到两个以上的块，本项目认为处于网络页面最上方的块是正文块，其他块是对正文块的回复。

当本项目抽取到多个正文块的时候，本项目一般会将其中所有的文本块抽取出来，然后依次拼接，组成一个大的文本。本项目成这个大文本就是本项目系统所要抽取的网络页面正文块。

在大量浏览分析相关网络页面源代码的基础上，本项目发现在网络页面的代码中，Script 代码以及一些 HTML 注释代码是用户不感兴趣的无效文本信息，在通过浏览器的调用，也不会显示在页面上的。本项目的目标有效文本块都是被包含在 “<DIV>”、“<BLOCKQUOTE>”、“<P>”、“<CENTER>”、“<H1>”到“<H6>”、“<SPAN>”和“<STRONG>”等标签内。这些块内的抽取到的文本信息一般都是有意义的，而且是完整的。

通过以上几个步骤，本项目就通过规则过滤，Tag 节点过滤再加上文本块拼接，找到了本项目的目标文本块。并将其抽取出来。

2.2.3 相关领域正文抽取技术试验分析

由于相关领域缺乏上标准的数据集，因此本实验的数据集是利用网络爬虫从各个不同的相关网站下载到的文本页面。在测试的时候，本项目必须雇用人工去查看页面的正文抽取情况，查看抽取结果是否分正确。然后通过一定的算法性能评价指标，最终评价出算法的好坏。

2.2.3.1 性能评价指标

本项目在本小节的实验中采用了精确度(*Precision*)、召回率(*Recall*)和 *F-Score* 来评价算法[49]。精确度是指所有被程序抽取出来的正确的正文块与所有程序抽取的 WEB 页面正文块之比例，召回率是指正确抽取的正文块与所有被人工抽取出来的 WEB 页面正文块的比例。

$$F-\mathrm{Score}=\frac{2\times Precision\times Recall}{Precision+Recall}$$

2.2.3.2 实验结果分析

首先，利用网络爬虫系统从相关网站爬取 200 个页面，总计 1200 个页面。将本项目实现的网络页面正文抽取算法应用到其中。本项目提出的算法，对这些常用的相关网站效果都很好。大部分正文抽取结果都在 85%以上。

2.4 本章小结

在本章主要描述安全领域搜索平台所涉及中的两个关键技术，面向爬虫的网络页面更新频率预测技术，该技术使用泊松分布来预测特定网络页面更新变化情况，为爬虫的准备爬行提供了基础。实验中本章使用的了三个评价标准（准确率、命中率和覆盖率）来判别本项目提出算法的性能，结果证明，本项目提出算法在 3 个评价标准上都超过 80%。在本章中我们还实现了面向相关领域的网络页面内容抽取技术。该技术利用网络页面结构化标记 DOM 树以及 HTML Tag 的视觉线索来抽取网络页面中最重要的内容。实验证明，本项目采用算法的网络页面正文抽取效率达到 85%。

3 平台的实现

3.1 爬虫模块实现

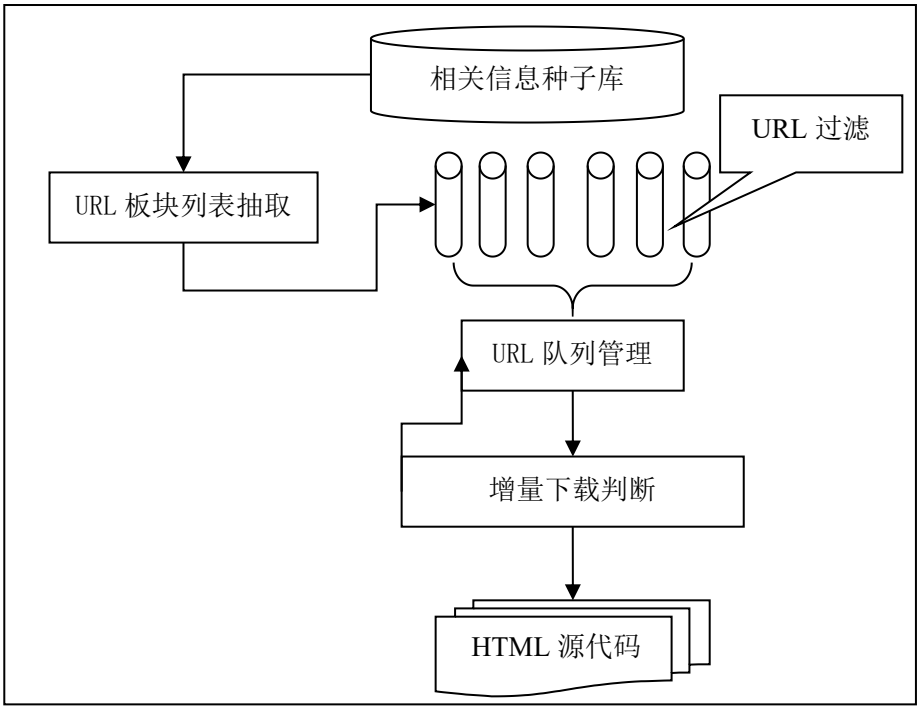

图 3-1 爬虫系统结构图

爬虫模块的输入和输出分别是相关信息领域种子库和 WEB 页面的 HTML 源代码文件。首先，数据库的种子信息被网络爬虫系统获取，这些种子信息包含了目标网站的入口 URL、以及一些网络爬虫的指导信息。种子信息被网络爬虫系统获取后，系统便开始下载种子所对应的列表界面，然后根据 URL 抽取器获取到主题的地址，并存放到系统内存队列当中。为了能够满足下载的时候各内存队列负载均衡，必须要有 URL 队列管理来处理这个过程。多线程下载在内存队列中有内容的时候开始工作，通过发送 HTTP 请求，将 URL 指定的目标页面的源代码下载下来。

下面本项目以新浪网为例，展示本项目提出的爬虫的设置过程：

(1) 增加种子：

可以通过按“增加”按钮来增加种子，本项目输入一个种子 http://www.sina.com.cn（选择的原因是该网站不需要登录用户名和密码等个人信息），以“域名”为种子类型，可以选择优先级为 1 到 10 的不同等级，此处本项目选择的是默认 5 级，后续都选择为默认等级。同步骤增加种子 http://www.163.com，此处本项目选择优先级为最高级 10 级。

(2)导入种子：

可以通过按“导入种子”导入已经存储在系统中的种子，本项目于处选择导入的种子为

http://www.sina.com.cn，选择优先级为最低级 1 级。

(3)导出种子：

通过按“导出全部种子”可以导出全部种子，按“导出选中的种子”来导出本项目需要保存的种子，此处本项目选择的是导出全部种子，保存为 test.xls，如图 3-2 所示：

图 3-2 爬虫系统设置种子

通过选中种子，可以进行对种子的操作。

(4)下载参数：

本项目选择的最大下载速度为 50KB/S，系统会在下载的速度大于 50KB/S 时自动调节降低下载的速度，这时可以在后台看到相应的信息提示。选择 50KB 为最大单个下载文件的大小，这项参数的设置限定了下载的网络页面不可能大于 50KB，可以在下载的页面中得到验证。本节涉及的各种设置可以由使用者根据自身的实际网络状况进行相应调整；域名的设置可以保持默认，也可以根据使用者自己的实际网络做出更改。代理服务器的设置可以设置为代理访问本项目网络访问受限的其他网络页面，需要设置的是代理服务器的端口号和地址，特别的代理服务器还需要输入代理服务器的用户和密码。具体如图 3-3 所示：

图 3-3 爬虫系统设置下载参数

(5)设置爬寻范围：

①扫描引擎：本系统中设置两种搜索引擎分别为 Irs 和 Dfa，可以不选择引擎也可以选择一种或两种扫描引擎，在本章节案例中选择的是不另外选择扫描引擎即默认方式。

②爬行层数的限定：这个参数指的是选择种子搜索扫描能够到达的层级（种子层为 0 层），使用者根据自身需求可以选择设定层数。本章节的案例中选择了 4 层，表示搜索扫描到的网络页面最多为 4 层，也可以在数据库 result 的数据项 Layor 中查看，最大层数不超过 4 层。如下图所示：

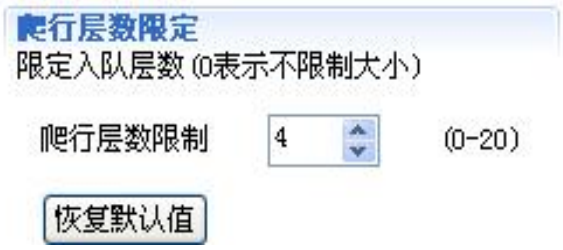

图 3-4 爬虫系统设置爬行层数

③IP 过滤：可以通过此项设置，设置本项目所需要的爬寻范围内必须排除或包括的 IP 或 IP 段，所以此范围内的 IP 在爬行的时候将不会爬寻，在 result 数据库中查询验证。如下图 3-5 所示：

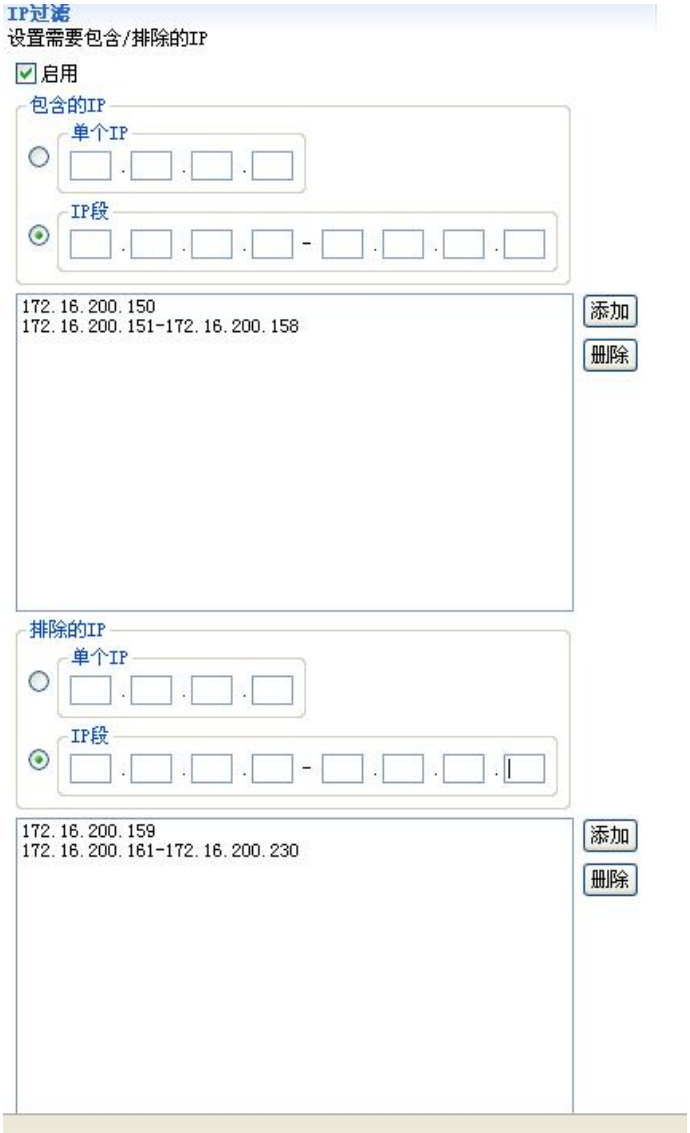

图 3-5 爬虫系统设置 IP 过滤

④站点过滤：可以在此处根据自己的需求来设定爬行范围所需要爬行的扩展域名，本案例设置如下图 3-6 所示：

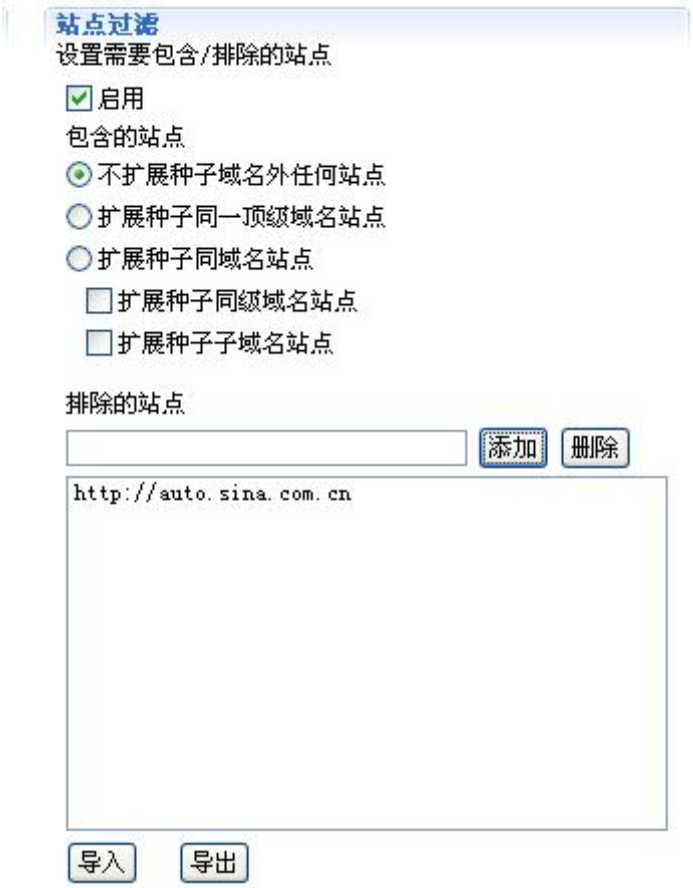

图 3-6 爬虫系统设置站点过滤

图中选择了“不扩展种子域名外任何站点”即爬寻的网络页面不会包含的站点为 http://auto.sina.com.cn。根据这些设置，本项目提出的爬虫就可以 24 小时不停的在服务器上运行，源源不断地为相关信息服务平台提供原始数据。

3.2 基于视觉线索的页面正文抽取模块的实现

网络页面正文抽取模块抽取的示意图如图 3-7 所示。该模块输入的单个网络页面的 HTML 代码，输出的网络页面正文。该模块中的正文识别算法有几个子算法组成，它们是分别是 tag 识别、DOM 结构识别、相关视觉信息识别、最小文本块识别、以及文本快合并。

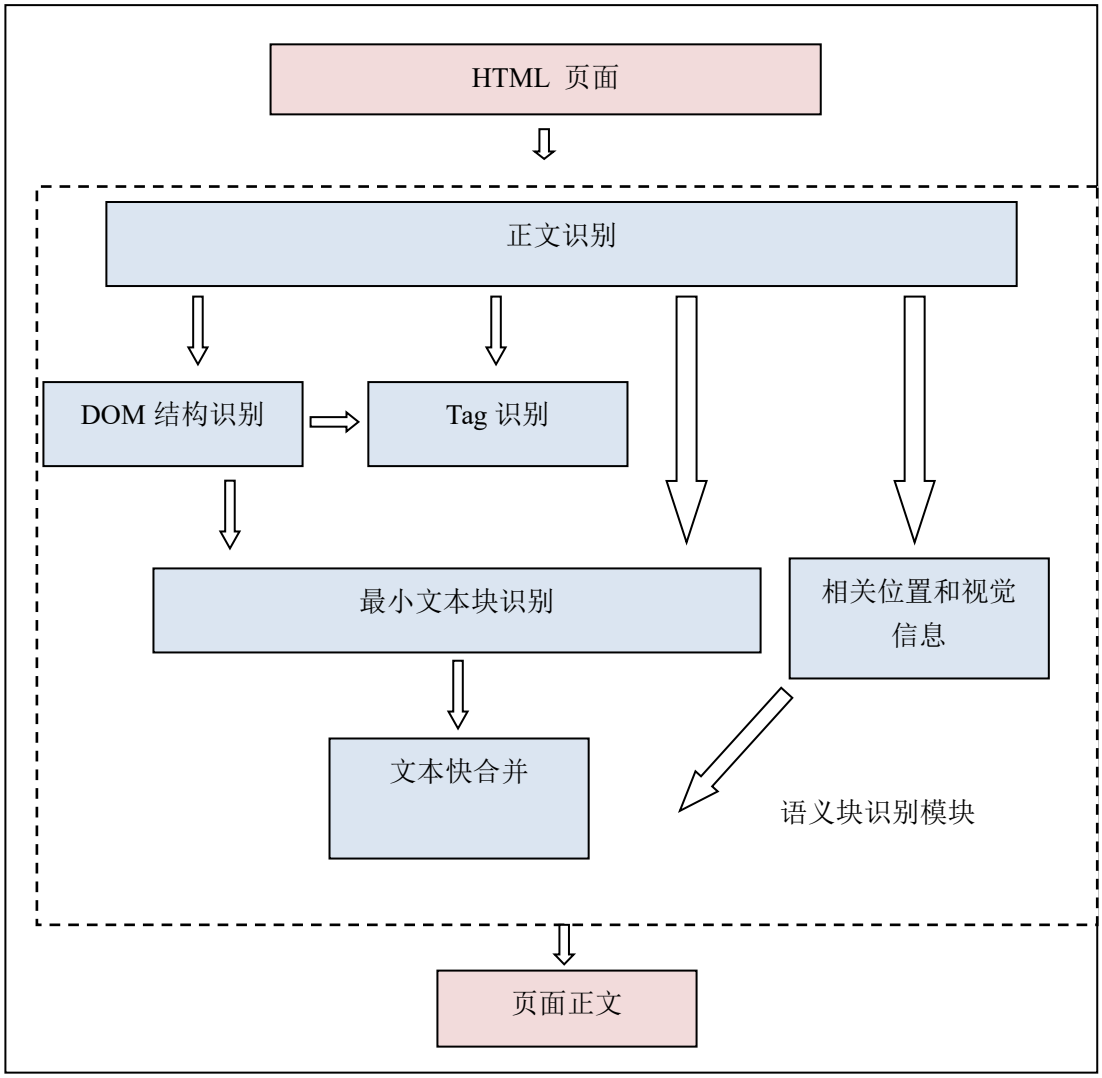

图 3-7 基于视觉信息的 WEB 页面正文抽取模块流程

每当一个 URL 被网络爬虫下载到本项目的系统中后，网络页面正文识别模块就会自动启动，将网络页面 HTML 代码转化为一棵 DOM 树。接着，本项目的程序将会遍历这棵 DOM，并将其中的结构 Tag 都识别出来。然后程序会将数中所有的最小文件块识别出来。在文本块识别的同时，程序还将识别该 tag 的视觉信息。在此基础上，本项目的程序，最后将得到整个网络页面的主要正文。

信息技术在相关信息服务领域的应用是目前国内外关注和研究的热点问题，平台正是把先进的互联网络技术与通讯技术应用到服务过程中，较好的解决了相关专业科学数据的监管服务问题。

4 结论

4.1 网络爬虫

到目前为止对网络爬虫访问策略的研究大致分成三个类型：（1）基于随机概率模型的访问策略，这类型的算法认为网络页面内容变化的频率服从随机的概率模型。很多研究调研了网络页面内容的速率，并对它进行了估计。使用了泊松分布、韦伯分布等的数据模型对网络页面的更新进行模拟，提出了不同的增量爬行策略。（2）基于样本采样的访问策略。这类型方法是最早由 Cho 和 Ntoulas 等人提出的。通常运行在独立的网站上，算法通过一个网站内一少量网络页面样本的变化情况来预估整个网站网络页面的变化情况。在这种情况下每出现一个网站，就需要重新训练一个新的概率模型。（3）基于分类模型的访问策略。这种类型方法使用网络页面及其内容更新的历史记录去预测网络页面的变化行为。算法根据网络页面的变化定义网络页面特征从而对网络页面进行分类。网络页面的每个类别代表不同的变化频率模型，需要单独训练。

本项目研究了基于泊松分布的网络页面更新频率预测算法，该算法预测了网络页面内容在未来的变化，解决了爬虫在什么时间增量爬行的关键问题。本算法还存在一些不足，需要在以后的研究中进一步扩展和深入的方向有两点：第一，基于泊松分布的网络页面更新预测算法目前是以假设所有的网络页面的更新变化都是在对等时间间隔内为前提的。那么泊松分布怎样对随机时间事件预测性行为延展到连续区间，这是需要继续深入研究的问题。第二，在网络页面的实际变化率在 0.4~0.6 范围内的时候，本项目提出的基于泊松分布的网络页面更新预测算法的预测效果不理想。因此，怎样提高在这个区间范围内网络页面变化的预测效果还要进一步的深入研究。

4.2 页面内容抽取算法

目前，网络页面内容的自动化抽取技术是信息检索领域的一个研究热点。我们研究的是相关网络页面内容的自动化抽取方法，实现了基于ＤＯＭ树结构和视觉信息的网络页面信息的自动抽取算法。通过分析大量相关内容网络页面的源代码和视觉信息，提出了一个结合网络页面ＤＯＭ树结构信息以及网络页面视觉信息的内容抽取算法。实验结果表明，采用这种方法，能够实现一个高效的，无需任何用户参与的全自动抽取器。实验证明这种方法在我们的目标网站是运行效果非常好，识别率达到９０％以上。本项目虽然取得了以上成果，但仍有很多不足之处，今后可以对以下几方面进行进一步的研究。第一，只研究了单个页面下的ＷＥＢ内容信息的识别抽取。在实验中一个长的文章一般被分割于多个内容模块，如何对多长文章进行有效的识别抽取将是以后的重点研究问题之一。第二，只研究了正文抽取技术，但是网络页面中一般还存在对网络页面正文的评价和回复。因此，如何对它们进行识别和抽取需要进行进一步的研究。第三，本项目研究的网络页面信息抽取技术提高了的目标是算法的准确性和可靠性，在时间效率上还有很大的提升空间。

5 部分软件现实代码

```
Pattern p = Pattern.compile(resultPattern,
                        Pattern.CASE_INSENSITIVE);
            Matcher m = p.matcher(page);
            if (m.find()) {
                String result = m.group(1);
                System.out.println("this search in google fetch "+result+ " records");
            }

        p = Pattern.compile(namePatternStr, Pattern.CASE_INSENSITIVE);
        m = p.matcher(page);
        while (m.find()) {
            String name = m.group(0).trim();
            name = name.substring(14, name.length() - 5);
            name = SearchStringUtil.normalizaitonResultString(name);
            names.add(name);
        }
        p = Pattern.compile(urlPatternStr, Pattern.CASE_INSENSITIVE);
        m = p.matcher(page);
        while (m.find()) {
            String url = m.group(0).trim();
            url = url.substring(6, url.length() - 7);
            url = SearchStringUtil.normalizaitonResultString(url);
            urls.add(url);
        }
        p = Pattern.compile(despPatternStr, Pattern.CASE_INSENSITIVE);
        m = p.matcher(page);
        while (m.find()) {
            String desp = m.group(0).trim();
            desp = desp.substring(17, desp.length() - 7);
            desp = SearchStringUtil.normalizaitonResultString(desp);
            deps.add(desp);
        }

        for(int i = 0; i < names.size(); i++){
            SearchResult sr = new SearchResult();
            sr.setDescription(deps.get(i));
            sr.setLink(urls.get(i));
            sr.setName(names.get(i));
            sr.setRankNum(i);
            ls.add(sr);
        }
        return ls;
    }

    public void close(){
```

```
            browserDownloader.close();
        }

        public static void main(String[] args){




        }
}

public class WebSiteCrawler {
        //parameters
        public int depth = 1 ;
        public int crawledCountThreshold = 3;

        public String homepageUrl = null;
        public DownLoader downloader = null;
        Set<String> crawlingURL = null;
        Set<String> finishedURL = null;

        public List<String> noTextFileFilter = null;

        public WebSiteCrawler(){
            initial();
            homepageUrl = "http://baidu.com";
            try {
                downloader = DownLoaderFactory.getDocDownloader("normal");
            } catch (IOException e) {
                e.printStackTrace();
            }

        }

        public WebSiteCrawler(DownLoader downloader){
            initial();
            homepageUrl = "http://baidu.com";
            this.downloader = downloader;
        }

      public WebSiteCrawler(String homepageUrl){
            initial();
            this.homepageUrl = homepageUrl;
            try {
                downloader = DownLoaderFactory.getDocDownloader("normal");
            } catch (IOException e) {
                e.printStackTrace();
            }
      }

      public WebSiteCrawler(String homepageUrl , DownLoader downloader){
```

```
        initial();
         this.downloader = downloader;
         this.homepageUrl = homepageUrl;
}

public WebSiteCrawler(String homepageUrl,String downloaderName){
        initial();
         this.homepageUrl = homepageUrl;
         try {
                downloader = DownLoaderFactory.getDocDownloader(downloaderName);
         } catch (IOException e) {
                e.printStackTrace();
         }
}


public void initial(){
        crawlingURL = new HashSet<String>();
        finishedURL = new HashSet<String>();
}
public String downloadPage(){
        String content = null;
         try {
                this.homepageUrl = UrlUtil.normalizeUrl4Crawler(this.homepageUrl);
                content = downloader.download(this.homepageUrl);

         } catch (IOException e) {
                System.out.println("failed in : homepage");
         }finally{
                if(null == content)
                        return null;
         }
         return content;
}


public String downloadPage(int count){
        String content = downloadPage();
        int tCount = 0;
         if(null == content && tCount < count){
                try {
                        content = downloader.download(this.homepageUrl);
                        tCount++;
                        try {
                                long s = (long) (1000 * Math.random());
                                Thread.sleep(s);
                        } catch (InterruptedException e) {

                        }

                } catch (IOException e) {
```

```
                System.out.println("failed in : homepage " + count);
            }finally{
                if(null == content)
                    return null;
            }
        }
        return content;
}

public String downloadPage(String u){
        String content = null;
        try {
            u = UrlUtil.normalizeUrl4Crawler(u);
            content = downloader.download(u);

        } catch (IOException e) {
            System.out.println("failed in : page " + u);
        }finally{
            if(null == content)
            return null;
        }
        return content;
}

public WebPage downloadPage(WebPage u){
        if(null == u.getUrl()){
        return null;
        }
        if(null != u.getContent()){
            return u;
        }else{
             u.setContent(downloadPage(u.getUrl()));
        }
       return u;
}

// according to the dcount a
public WebPage downloadPage(WebPage u, int dCount){
       if(null == u.getUrl()){
         return null;
       }

       while(!u.isCrawled()){

         if(null != u.getContent()){
            return u;
         }else{
             String c = downloadPage(u.getUrl());
             if(null != c ){
                 u.setCrawled(true);
                 u.setContent(c);
```

```
                }else{
                        u.setCrawledCount(u.getCrawledCount() + 1);
                        if(u.getCrawledCount() >= dCount -1){
                                u.setCrawled(true);
                        }
                }
        }
        }
      return u;
}

 /**
  * if a element in return list is null, show that the page can't be downloaded.
  * @param urls
  * @return
  */

 public List<String> downloadPage(List<String> urls){
              List<String> contents = new ArrayList<String>();
               for(int i= 0 ; i < urls.size(); i++){
               String content = downloadPage(urls.get(i));
               contents.add(content);
               }

        return contents;
 }
 /**
  * get the page content in term of url list
  * @param urls: url list
  * @param dCount: the number of trying without successful download
  * @return
  */
 public List<WebPage> downloadPage(List<String> urls, int dCount){
        List<WebPage> crawlingPages = new ArrayList<WebPage>();
        List<WebPage> finishedPages = new ArrayList<WebPage>();
        for (int i = 0; i < urls.size(); i++) {
                WebPage page = new WebPage(urls.get(i));
                crawlingPages.add(page);
                DownLoader.randSleep(1000);
                downloadPage(page, dCount);
        }
          List<WebPage> remove = fileterCrawledPage(crawlingPages, finishedPages);
        while (0 < crawlingPages.size()) {

                for (int i = 0; i < crawlingPages.size(); i++) {

                        downloadPage(crawlingPages.get(i),dCount);
                        DownLoader.randSleep(1000);
                }
                fileterCrawledPage(crawlingPages, finishedPages);
        }
```

参考文献

[1] 刘敏,“双一流”高校图书馆科学数据服务现状及优化策略[J].图书馆工作与研究 2020，11：15-23

[2] 叶兰，高校图书馆科学数据管理服务能力成熟度模型构建与应用研究 [J]. 图书馆情报工作，2022,3:3-14

[3] 国务院办公厅印发《科学数据管理办法》[EB/ OL]．[2021-10-14]．http：//www.gov.cn/home/2018-04/ 02/content_5279296.htm.

[4] 《科研数据北京宣言》[EB/OL]．[2021-10-15]．http：//www.codata.org/uploads/Beijing%20Declaration- 19-11-07-FINAL.pdf.

[5] 国家科学数据中心主任第三次联席会议召开 [EB/OL]．[2021-10-13]．http：//www.most.gov.cn/kjbgz/ 202104/t20210426_174219.html.

[6] CASTLE C.Getting the central RDM message across：A case study of central versus discipline-specific research data services（RDS）at the University of Cambridge [J].Libri，2019，69（2）：105-116.

[7] BERNDSEN C.Teaching data management and liter- acy to support course embedded research projects [J].The FASEB Journal，2021，35.

[8] 孙丹霞，王伟军，姜毅.基于用户小数据的嵌入式 学科服务研究 [J].图书馆工作与研究，2019（4）：84-90.

[9] 吴爱芝，王婧媛.大数据时代高校图书馆嵌入式 科研服务模式与内容研究 [J].现代情报，2018，38（12）： 97-102.

[10] 王艳敏.英国罗素大学集团成员图书馆慕课资源 嵌入式服务研究[J].图书馆工作与研究，2021（9）：53- 61.

[11] INUWA S，ABRIZAH A.Embedded librarianship in research in Nigerian Universities：Practices and sources of practice knowledge [J].The Journal of Academic Librarian- ship，2018，44（6）：738-746.

[12] AL-JARADAT O M.Research data management （RDM）in Jordanian public university libraries：Present status，challenges and future perspectives [J].The Journal of Academic Librarianship，2021，47（5）：102378.

[13] 徐菲，王军，曹均，等.康奈尔大学嵌入式科研数 据管理服务探析 [J].图书馆建设，2015（12）：54-59.

[14] 尹春晓.高校科学数据管理嵌入式服务模式探 索[J].情报资料工作，2017（2）：77-82.

[15] 金贞燕，阿童木.科研数据管理服务内容体系构 建研究 [J].情报理论与实践，2021，44（8）：42-50.

[16] Kang C, DOM-based Web Pages to Determine the Structure of the Similarity Algorithm[C]. Intelligent Information Technology Application, 2009. IITA 2009. Third International Symposium, 2009, 2:245-248.

[17] Liu R, Xiong R, Gao K, Web Object Block Mining Based on Tag Similarity[C]. 2012 International Conference on Intelligent Computation Technology and Automation (ICICTA), 2012, 3:1159-1162.

[18] 聂卉, 黄贵鹏. 树编辑距离在 Web 信息抽取中的应用与实现[J], 现代图书情报技术, 2010(5):29-34.

[19] Reis D C, Golgher P B, Silva A S, et al. Automatic Web News Extraction Using Tree Edit Distance[C]. Proceedings of the 13th international conference on World Wide Web, 2004:502-511.

[20] Cai D, Yu S, Wen J R, et al. Extracting Content Structure for Web Pages based on Visual Representation[C]. Proceedings of the 5th Asia-Pacific web conference on Web technologies and applications, 2003:406-417.

[21] Cai D, Yu S, Wen J R, et al. Vips: a Vision Based Page Segmentation Algorithm[R]. Microsoft Technical Report. MSR-TR-2003-79, 2003:10.

[22] Yu S, Cai D, Wen J R, et al. Improving Pseudo-Relevance Feedback In Web Information Retrieval Using Web Page Segmentation[C]. Proceedings of the 12th International Conference on World Wide Web, 2003:11-18.

[23] Mehta R R, Mitra P, Karnick H, Extracting Semantic Structure of Web Documents Using Content and Visual Information[C]. Special Interest Tracks and Posters of the 14th International Conference on World Wide Web, 2005:928-929.

[24] Liu W, Yan H L, Yang J W, et al. A Unified Approach for Extracting Multiple News Attributes from News Pages[C]. Proceedings of the 11th Pacific Rim International Conference on Trends in Artificial Intelligence, 2010:157-169.

[25] Liu W, Yan H L, Xiao J G, et al. Extracting multiple news attributes based on visual features[J]. Journal of Intelligent Information Systems, 2012, 38(2):465-486.

[26] Liu W, Yan H L, Xiao J G, et al. Extracting multiple news attributes based on visual features[J]. Journal of Intelligent Information Systems, 2012, 38(2):465-486.

[27] Wang J F, Chen C, Wang C, et al. Can We Learn a Template-Independent Wrapper for News Article Extraction from a Single Training Site[C]. Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009:1345-1353.

[28] Flesca S, Manco G, Masciari E, et al. Web Wrapper Induction: a brief survey[J]. AI Commun, 2012, 17(2):57-61.

[29]Crescenzi V, Mecca G, Merialdo P. RoadRunner: Towards Automatic Data Extraction from Large Web Sites[C]. Proceedings of the 27th International Conference on Very Large Data Bases, 2001:109-118.

[36] Dalvi N, Kumar R, Soliman M. Automatic Wrappers for Large Scale Web Extraction[J].

Proceedings of the VLDB Endowment, 2011, 4(4):219-230.
[37] Cardoso E, Jabour L, Laber E. An Efficient Language-Independent Method to Extract Content from News Webpages[C]. In Proceeding of Proceedings of the 2011 ACM Symposium on Document Engineering, 2011:121-127.
[38] 张俊英, 胡侠, 卜佳俊. WEB 页面文本信息自动提取技术综述[J]. 计算机应用研究, 2009, 26(8):2827-2831.
[39] Chung C Y, Gertz M, Sundaresan N. Reverse Engineering for Web Data: From Visual to Semantic Structures[C]. Proc of the 18th International Conference on Data Engineering, 2002:53-63.
[40] Zhong P, Chen J. A Generalized Hidden Markov Modelapp Roach for Web Information Extraction[C]. Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, 2006:709-718.
[41] Peshkin L, Pfeffer A. Bayesian Information Extraction Network[C] Proceedings of the 18th International Joint Conference on Artificial Intelligence, 2003:421-426.
[42] Ray S, Craven M. Representing Sentence Structure in Hidden Markov Models for Information Extraction[C]. Proceedings of the 17th International Joint Conference on Artificial Intelligence, 2001, 2:1273-1279.
[43] Zhu J, Nie Z, Wen J R, et al. Simultaneous Record Detection and Attribute Labeling in Web Data Extraction[C]. Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006: 494-503.
[44] Wu X, Kumar V, Ross Q J, et al. Top 10 Algorithms in Data Mining[J]. Knowledge and Information Systems, 2008, 14(1):1-37.
[45] Quinlan J R. Induction of Decision Trees. Machine Learning[J], 1986, 1(1):81-106.
[46] Quinlan J R. C4.5:Programs for Machine Learning[M]. Morgan Kaufmann Publishers, 1993.
[47] Riber S, Marathe U P, The Single Pass Clustering Method[R]. In Report ISR-16 to the National Science Foundation, Cornell University, Department of Computer Science, 1969.
[48] Pasternack J, Roth D. Extracting Article Text from the Web with Maximum Subsequence Segmentation[C]. Proceedings of the 18th International Conference on World Wide Web, 2009:971-980.
[49] Fan J, Luo P, Joshi P. Title Identification of Web Article Pages Using HTML and Visual Features[C]. Proceedings of SPIE-IS&T Electronic Imaging, 2011, 7879:78790K1-78790K5