信息内容安全实验技术报告

项目名称：微博的内容识别与控制

韦昆杰

1181000420

348971397@qq.com

2021 年 6 月 23 日

目录

1 摘要

本实验主要实现了一个小型的信息安全管理系统，其中包含以下功能：

1.1 网络信息数据分类捕获

1. 通过爬虫对微博进行爬取，得到 HTML
2. 通过对 HTML 的进一步提取分析，得到信息

1.2 网络协议还原分析

1. 实现了常见的字符串匹配算法 (BF KMP AC)
2. 当关键字为一个时采用 KMP 算法匹配，当关键字为多个时采用 AC 算法匹配
3. 通过将获取的信息进行关键字匹配，可以识别包含指定关键字的页面

1.3 网络信息内容或报文特征识别

1. 采用主流的字符串匹配算法，匹配其中的主题相关关键词。
2. 还可以进行离线的分析，得到主题的统计结果

1.4 网络信息安全管理响应

1. 针对前面匹配到的关键字或者网站地址，设计并实现有效的控管操作
2. 采用水军的引导方法，使得某微博被淹没
3. 通过举报匹配到的微博使其被删除

1.5 系统辅助功能部分

1. MongoDB 数据库存储爬取的热搜榜以及微博信息

2. 实现了网站页面，用户可以通过 GUI 完成各项操作

本系统在 Windows10 平台上进行开发和测试，使用的开发环境为 Visual Studio Code，开发语言为 TypeScript，使用的数据库为 MongoDB 数据库，网站页面使用 HTML CSS 实现，网站爬取采用 puppeteer 实现。

2 项目简介

2.1 背景分析

了解信息安全管理系统设计的关键环节，掌握网络信息内容的高效的数据捕获技术、准确识别及分析各种常用协议报文，并可针对用户的屏蔽内容需要，拦截或统计分析相应的特征报文。能够针对信息内容安全领域的新进展、新应用，设计与开发实用化的网络信息监控系统。

2.2 特色描述

通过主动爬虫爬取微博，得到微博信息，存储到数据库中。然后设置关键字，通过匹配算法进行匹配，并对匹配到的微博进行管控 (水军或举报)

3 实现方案

3.1 系统方案

系统平台: Windows 10
开发语言: TypeScript
界面实现: HTML CSS
数据库系统: MongoDB 数据库

3.2 实现原理

3.2.1 微博热搜子系统

实时爬取微博当前的热搜，并在网页端展示出来，并存储到数据库中

3.2.2 微博搜索子系统

输入要搜索的内容，系统爬取相关微博展示，爬取到的所有微博信息存储到数据库中

3.2.3 微博过滤子系统

设置关键字，系统采用 KMP 或 AC 算法对微博进行匹配，匹配到的微博展示到网页上

3.2.4 微博管控子系统

对匹配到的微博，可以进行管控，通过举报该微博或进行相关信息微博的不断发生淹没该微博

3.2.5 数据库子系统

这里借助 MongoDB 数据库，设计数据库操作子系统，将微博信息存储到数据库中

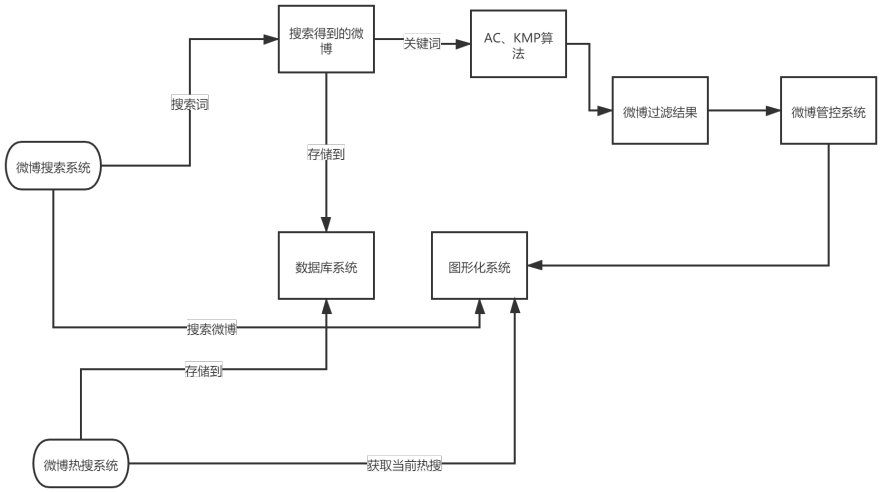
3.2.6 图形化界面子系统

利用 HTML CSS 设计网页 UI，TypeScript 实现动态交互

3.3 硬件框图

本系统不涉及硬件结构

3.4 系统软件流程



4 系统功能设计

4.1 微博热搜子系统

该系统通过 HTTP 请求热搜网页，得到对应的 HTML 后对其进行进一步的提取分析，得到微博热搜编号、内容、搜索量信息，将这些信息存储到数据库中并且展示在页面上

4.2 微博搜索子系统

该系统接受用户搜索，通过 HTTP 命令得到搜索对应的微博 HTML，解析 HTML 得到各条微博的 URL，通过 Puppeteer 模拟分别爬取各条微博的内容。得到微博 URL、发送用户、微博发送时间、微博内容、点赞数、评论数、转发数等信息存储到数据库中，并展示在页面上

4.3 微博过滤子系统

该系统接受用户输入关键词，系统根据关键词个数 (如果为 1 采用 KMP 算法，否则采用 AC 算法) 进行匹配，匹配后的结果展示在页面上

4.4 微博管控子系统

用户可以选择是否对过滤到的微博进行管控，如果选择管控，系统会通过 Puppeteer(需要一个登录的账号，模拟真实的微博用户) 对相应微博进行举报或者产生大量相同内容微博

4.5 数据库子系统

数据库采用 MongoDB，对收集到的热搜信息和微博信息分别存储到相应的 Collection 中，可以随时查看

4.6 图形化界面子系统

本实验采用网页，通过 HTML CSS 设计网页 UI，TypeScript 实现动态交互

5 关键技术

5.1 KMP 算法

当关键词个数为 1 时，微博过滤子系统采用 KMP 算法对微博内容进行匹配。

KMP 算法首先通过计算模式串的最长公共前后缀，并将结果存储到表中，并根据该表进行匹配。一个词在不匹配时本身就包含足够的信息来确定下一个匹配可能的开始位置，此算法利用这一特性以避免重新检查先前匹配的字符。

该算法时间复杂度为 O(m)+O(n)，空间复杂度为 O(m)

5.2 AC 算法

当关键词个数超过 1 个时，微博过滤系统采用 AC 算法对微博内容进行匹配。

AC 算法在初始阶段，建立三个函数，转向函数 goto，失效函数故障和输出函数输出，在搜索查找中，通过这三个函数的交叉使用扫描，定位出关键字在文本中的所有出现位置

该软件扫描文本时完全无需回溯，时间复杂度为 O(n)，时间复杂度与关键字的数量和长度无关系

5.3 React

React（有时叫 React.js 或 ReactJS），是一个为数据提供渲染为 HTML 视图的开源 JavaScript 库。React 视图通常采用包含以自定义 HTML 标记规定的其他组件的组件渲染。React 为程序员提供了一种子组件不能直接影响外层组件 (data flows down) 的模型，数据改变时对 HTML 文档的有效更新，和现代单页应用中组件之间干净的分离。

本实验采用 React，通过 JSX(JavaScript XML) 描述页面结构，最终渲染为 HTML

5.4 Puppeteer

Puppeteer 是一个 Node 库，它提供了一个高级 API 来通过 DevTools 协议控制 Chromium 或 Chrome。

本实验采用 Puppeteer，模拟微博用户实现微博爬取和微博举报等功能

6 总结

本实验主要实现了一个小型的信息安全管理系统，实现了对动态网页 (微博网页版) 的内容识别与控制

7 参考文献

- KMP 算法-维基百科 [https://zh.wikipedia.org/wiki/KMP 算法]
- AC 算法-维基百科 [https://zh.wikipedia.org/wiki/AC 自动机算法]
- React-维基百科 [https://zh.wikipedia.org/wiki/React]
- Puppeteer [https://pptr.dev/]