****

**本科毕业设计**

**（2018届）**

|  |  |
| --- | --- |
| **题 目** | **个性化新闻推荐系统** |
| **学 院** | 计算机学院 |
| **专 业** | 物联网工程 |
| **班 级** | 14058911 |
| **学 号** | 14051336 |
| **学生姓名** | 张慧 |
| **指导教师** | 李珣 |
| **完成日期** | 2018 年 6 月 |

**诚 信 承 诺**

我谨在此承诺：本人所写的毕业论文《个性化新闻推荐系统》均系本人独立完成，没有抄袭行为，凡涉及其他作者的观点和材料，均作了注释，若有不实，后果由本人承担。

**承诺人（签名）：**

**年 月 日**

# 摘要

在当下信息发达的网络时代，网络新闻应用作为人们获取网络新闻的主要渠道，已经成了人们生活不可或缺的一部分，它的使用率一直保持在较高水平。新闻的发展经历了传统媒体时代、PC互联网时代以及移动互联网时代，在移动互联网时代兴起了个性化新闻推荐。当下新闻入口呈现多样化特点，新闻内容丰富多样，新闻数据海量。个性化新闻推荐系统通过采集海量信息以及数据挖掘，智能分析出热点资讯，然后根据用户的社交关系链和浏览记录等建立用户模型，对用户的信息偏好进行预判，实现向用户智能推荐个性化的信息。个性化的新闻推荐手段一定程度上满足了用户的需求，提升了用户体验。

个性化的推荐相对于大而全的信息提供方式是一种巨大的飞跃。在大数据时代下，个性化推荐技术使人们在丰富且略显庞杂的信息社会中获取信息变得快速与高效，降低了信息检索的时间与精力成本。个性化推荐技术的核心是个性化推荐算法，当下的个性化推荐算法多样有基于内容的推荐（Content-based Recommendation）、基于用户的协同过滤(user-based CF)、基于项目的协同过滤(item-based CF)、SVD、基于深度学习的推荐算法、混合推荐等算法。结合深度学习的大数据推荐系统已是一种发展趋势。

本文围绕个性化新闻推荐的设计与实现展开，讨论了个性化推荐的基本算法。本文实现的个性化新闻推荐系统是以Spring+SpringMVC+Mybaties整合框架（SSM框架）为开发平台，通过爬虫技术爬取网络新闻，由推荐模块计算推荐结果，最后由web展示结果，用户使用本系统将会产生用户相关数据。

**关键词：** 协同过滤；个性化；新闻推荐；SVD；大数据；SSM

# ABSTRACT

In the current information-rich network era, the application of network news, as the main channel for people to obtain network news, has become an indispensable part of people's life, and its utilization rate has been kept at a high level. The development of news has gone through the traditional media era, the PC Internet era and the mobile Internet era. In the mobile Internet era, personalized news recommendation has emerged. At present, the news portal has diversified features, rich and varied news contents, and huge amounts of news data. By collecting mass information and data mining, personalized news recommendation system can analyze hot information intelligently, and then establish user model according to the user's social relationship chain and browsing record, and prejudge the user's information preference, so as to recommend personalized information to the user intelligence. Personalized news recommendation methods satisfy the needs of users to a certain extent and enhance user experience.

Personalized recommendation is a huge leap compared to the large and complete information provision. In the era of large data, personalized recommendation technology makes people get information quickly and efficiently in rich and slightly miscellaneous information society, and reduces the time and energy cost of information retrieval. The core of personalized Recommendation technology is personalized Recommendation algorithm. Currently, there are various personalized Recommendation algorithms, including content-based Recommendation, user-based CF, item-based CF, SVD, Recommendation algorithm based on in-depth learning, hybrid recommendation algorithm and other algorithms. The big data recommendation system combined with in-depth learning has become a development trend.

This paper focuses on the design and implementation of personalized news recommendation, and discusses the basic algorithm of personalized recommendation. The personalized news recommendation system implemented in this paper is based on the spring + spring MVC + mybates integration framework as a development platform. through crawler technology, web news is crawled, recommendation results are calculated by the recommendation module, and finally displayed by the web. users using this system will generate relevant user data.

**Key words:** collaborative filtering; Personalized; News recommendation; SVD; Big data; SSM

目录

[摘要 3](#_Toc515990917)

[ABSTRACT 4](#_Toc515990918)

[目录 5](#_Toc515990919)

[1.引言 7](#_Toc515990920)

[1.1项目背景 7](#_Toc515990921)

[1.2发展历程 7](#_Toc515990922)

[1.3研究现状 8](#_Toc515990923)

[1.4基本工作内容 8](#_Toc515990924)

[1.5文本组织结构 8](#_Toc515990925)

[2. 相关技术 10](#_Toc515990926)

[2.1 SSM框架 10](#_Toc515990927)

[2.1.1 spring框架 10](#_Toc515990928)

[2.1.2 SpringMVC框架 11](#_Toc515990929)

[2.1.3 Mybatis框架 12](#_Toc515990930)

[2.2 存储相关 13](#_Toc515990931)

[2.2.1 MySQL数据库 13](#_Toc515990932)

[2.2.2 Mongodb数据库 13](#_Toc515990933)

[2.2.3 Ehcache缓存 13](#_Toc515990934)

[2.3爬虫技术 14](#_Toc515990935)

[2.4 相关框架 15](#_Toc515990936)

[2.4.1任务调度框架Quartz 15](#_Toc515990937)

[2.4.2 消息队列框架ActiveMQ 15](#_Toc515990938)

[2.5 前端技术 16](#_Toc515990939)

[2.6 本章小结 17](#_Toc515990940)

[3. 推荐算法研究 18](#_Toc515990941)

[3.1 基于协同过滤(Collaborative Filtering)的推荐 18](#_Toc515990942)

[3.1.1 基于用户的协同过滤算法（userCF） 18](#_Toc515990943)

[3.1.2 基于项的协同过滤算法（itemCF） 20](#_Toc515990944)

[3.2基于SVD的推荐算法 21](#_Toc515990945)

[3.2.1特征值 21](#_Toc515990946)

[3.2.2 SVD 22](#_Toc515990947)

[3.2.3奇异值分解SVD推荐算法 23](#_Toc515990948)

[3.3奇异值分解SVD与协调过滤的结合 24](#_Toc515990949)

[3.4本章小结 24](#_Toc515990950)

[4. 需求分析 25](#_Toc515990951)

[4.1需求概述 25](#_Toc515990952)

[4.2功能性需求 25](#_Toc515990953)

[4.2.1爬虫系统 26](#_Toc515990954)

[4.2.2新闻处理功能 27](#_Toc515990955)

[4.2.3用户模块功能 27](#_Toc515990956)

[4.2.4日志处理功能 28](#_Toc515990957)

[4.2.5消息处理 28](#_Toc515990958)

[4.2.6任务调度 28](#_Toc515990959)

[4.2.7评分系统 28](#_Toc515990960)

[4.2.8推荐处理 29](#_Toc515990961)

[4.3实体-关系图 29](#_Toc515990962)

[4.4非功能性需求 31](#_Toc515990963)

[4.5本章小结 31](#_Toc515990964)

[5. 系统设计与实现 32](#_Toc515990965)

[5.1系统架构 32](#_Toc515990966)

[5.2数据库设计 33](#_Toc515990967)

[5.3环境说明 35](#_Toc515990968)

[5.4实现说明 36](#_Toc515990969)

[6. 系统测试 37](#_Toc515990970)

[7. 总结与展望 38](#_Toc515990971)

[参考文献 39](#_Toc515990972)

[图表公式目录 41](#_Toc515990973)

[源码附录 41](#_Toc515990974)

# 1.引言

## 1.1项目背景

随着互联网的高速发展和移动设备的快速普及，网络新闻已经成为最重要的网络应用之一，根据CNNIC第40次中国互联网络发展状况统计，截至 2017 年 6 月，我国网络新闻用户规模为 6.25 亿，半年增长率为 1.7%，网民使用比例为 83.1%。其中，手机网络新闻用户规模达到 5.96 亿，占手机网民的 82.4%，半年增长率为 4.4%。

我们正处于大数据时代，每天在我们不经意的每分每秒，互联网上已经出现了海量的信息，网络新闻伴随时代的发展也呈现出海量式的数据。如此庞大的新闻数据为用户提供了各式各样的信息需求，但是对于用户来说精确的找到自己所需的信息，却变成一件难事，用户需要花时间与精力在提供的信息中二次查找，这严重影响了用户对新闻信息的获取和阅读体验。针对用户对于海量信息的问题我们称之为了信息过载（Information Overlord）问题即互联网用户在将大量的信息转化的过程中存在着“瓶颈”和“障碍”，无法将其内化为自己所需知识而加以有效吸纳，导致了“过载”。

针对信息过载问题，目前主要的解决方案是信息检索技术和个性化推荐技术。以搜索引擎为代表的信息检索技术，用户可以通过关键字检索获取自己所需信息，但同一关键字搜索的结果是一致的，无法满足个性化的信息需求。而个性化推荐系统很好地解决了这一问题，它依据用户的个性化需求、兴趣爱好等，使用不同的推荐算法，自动将用户感兴趣的信息和产品推荐给用户，为用户提供个性化的服务，同时也能增加用户依赖，提高用户忠诚度。

个性化新闻推荐系统不仅缘于大数据时代，也缘于大数据技术的发展。Hadoop、Spark等技术的出现为个性化新闻推荐提供了技术可行性。在大数据技术框架下，软件后台可以收集用户的网上痕迹从而分析用户偏好，根据分析结果对新闻信息进行精确推送。

## 1.2发展历程

推荐经历了由非个性化推荐到个性化推荐的过程。自1995年3月，在美国 人工智能协会上，卡耐基.梅隆大学的Robert Armstrong等人提出了个性化导航系统Web Watcher以及斯坦福大学的Marko Balabanovic等人推出了个性化推荐系统LIRA后，个性化推荐技术进入人们视野，随后有许多应用个性化推荐的系统诞生。个性化推荐技术在当下我认为已经是相当成熟了，当下个性化推荐算法众多有基于内容的推荐（Content-based Recommendation）、基于用户的协同过滤(user-based CF)、基于项目的协同过滤(item-based CF)、基于模型的协同过滤、基于深度学习的推荐算法、混合推荐等算法，这些算法各有优劣，必须结合实际合理选择。

个性化推荐应用最广的是电商，相较电商网络新闻应用个性化推荐就比较晚。现今个性化新闻推荐系统众多国外有News360、CNN的Zite、Flipboard、Pulse、Yahoo的digest等，国内有网易新闻、腾讯新闻、新浪以及以个性化新闻推荐出名的今日头条、天天快报、一点资讯等，新闻的个性化推荐已是各大新闻平台必备的功能，各大新闻平台也在努力完善推荐系统，力求为用户提供精准化推荐。

## 1.3研究现状

目前协同过滤的推荐算法、基于内容的推荐、混合推荐等几种比较常用的推荐算法已广泛地应用于包括电子商务在内的各种领域，极大地提升了用户体验。但也存在着不少问题，例如数据稀疏性问题、冷启动问题、可扩展性问题等。随着近几年深度学习的快速发展以及在多领域的应用为推荐系统提供了一种新思路，可能利用深度学习的方法能够解决传统推荐算法中面临的问题。

经过多年的发展，推荐技术取得了长足的进步。随着应用领域的不断拓展，将其他技术与推荐技术相结合将是一种趋势，尤其是人工智能与推荐技术的融合方向，它将具有十分广阔的应用场景。推荐系统技术的发展离不开各种困难和挑战，我们要不断开拓进取，使推荐技术朝着更智能，更优化的方向发展。

## 1.4基本工作内容

本文将围绕个性化新闻推荐的设计与实现展开。本系统的设计目标是利用奇异值分解SVD与协调过滤算法，根据用户历史行为构建一个用户新闻兴趣模型，实现一个个性化新闻推荐阅读平台。工作的主要内容如下：

1. 软件工程理论与关键技术算法研究：研究个性化推荐的相关算法，以及系统开发过程相关理论。
2. 系统整体框架设计及开发准备工作：分析系统需求，完成架构设计，根据需求分析搭建开发环境。
3. 系统原型完成：本系统采用B/S模式，使用SSM框架，结合WebMagic、ActiveMQ、Quartz、Ecache开发后端，数据库使用MySQL、Mongodb，前端使用AngularJS、angular Material、bootstrap、JSON技术，前后端使用RESTful API交互。
4. 系统测试：对设计的模块功能进行测试，并根据测试中遇到的问题对系统进行修复与调整。

## 1.5文本组织结构

论文主要分为七个章节，各章节内容如下。

第一章 引言。介绍个性化新闻推荐系统的背景，研究的现状，本人的工作，文本组织。

第二章 相关技术。介绍开发时使用的相关技术架构。

第三章 个性化推荐算法研究。介绍对个性化推荐算法的学习了解，主要是对协同过滤算法和奇异值分解SVD算法的介绍。

第四章 需求分析。对个性化新闻推荐系统进行需求概述。

第五章 系统设计与实现。对模块功能进行说明，以及开发环境、开发细节说明。

第六章 系统测试。介绍系统测试使用的框架

第七章 总结与展望。对文本的总结与个性化新闻推荐的展望。

# 2. 相关技术

本结主要是对开发个性化新闻推荐系统相关技术的介绍。

## 2.1 SSM框架

SSM框架主要是指的spring、springMVC、mybatis三大框架整合的框架。

### 2.1.1 spring框架

Spring是一个开源框架，Spring是于2003 年兴起的一个轻量级的Java 开发框架，由Rod Johnson创建。

Rod Johnson在2002年编著的《Expert one on one J2EE design and development》一书中，对Java EE 系统框架臃肿、低效、脱离现实的种种现状提出了质疑，并积极寻求探索革新之道。以此书为指导思想，他编写了interface21框架，这是一个力图冲破J2EE传统开发的困境，从实际需求出发，着眼于轻便、灵巧，易于开发、测试和部署的轻量级开发框架。Spring框架即以interface21框架为基础，经过重新设计，并不断丰富其内涵，于2004年3月24日，发布了1.0正式版。

Spring框架可以说是Java开发人员使用的最流行的应用程序开发框架之一。它目前由大量提供一系列服务的模块组成。包括模块容器，为构建横切关注点提供支持的面向切面编程（AOP），安全框架，数据存取框架，Web应用框架和用于模块测试提供支持的类。Spring框架的所有组件都通过依赖注入粘在一起。依赖注入（也称为控制反转）使设计和测试松散耦合的软件模块变得更容易。 

图 1 Spring 框架总览

### 2.1.2 SpringMVC框架

springMVC是spring系列中的一个开源项目。Spring MVC 是一个模型-视图-控制器（MVC）的Web框架建立在中央前端控制器servlet（DispatcherServlet），它负责发送每个请求到合适的处理程序，使用视图来最终返回响应结果的概念。Spring MVC是Spring产品组合的一部分，它享有 Spring IoC容器紧密结合Spring松耦合等特点，因此它有Spring的所有优点。

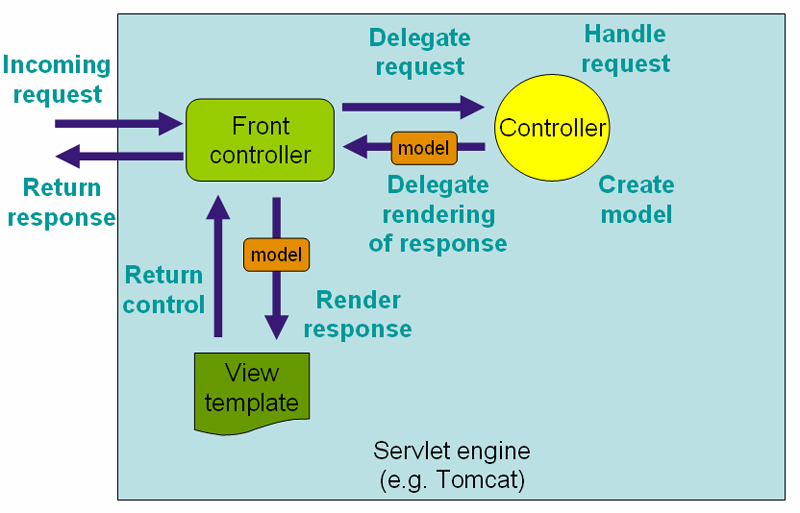


图 2 SpingMVC框架

SpringMVC的工作原理如下

1. 用户请求由前置控制器DispatcherServlet拦截。
2. DispatcherServlet调用HandlerMapping处理映射器。
3. HandleMapping根据xml配置或注解找到具体的处理器，生成处理器对象及处理器拦截器最后返回给DispatcherServlet。
4. DispatcherServlet调用HandlerAdapter处理适配器。
5. HandlerAdapter经过适配调用具体的处理器(Controller，也叫后端控制器)。
6. Controller执行完成返回ModelAndView。
7. HandleAdapter将Controller的执行结果ModelAndView返回给DispatchServlet
8. DispatcherServlet将ModelAndView传给ViewReslover视图解析器。
9. ViewReslover解析后返回具体的View。
10. DispatcherServlet根据View进行渲染视图（即将模型数据填充至视图中）。
11. DispatcherServlet响应用户请求。

### 2.1.3 Mybatis框架

MyBatis的前身是apache的一个开源项目iBatis，2010年这个项目由apache software foundation 迁移到了google code，并且改名为MyBatis。MyBatis是一个数据持久层(ORM)框架。

MyBatis 是支持普通 SQL查询，存储过程和高级映射的优秀持久层框架。MyBatis 消除了几乎所有的JDBC代码和参数的手工设置以及结果集的检索。MyBatis 使用简单的XML或注解用于配置和原始映射，将接口和 Java 的POJOs（Plain Old Java Objects，普通的 Java对象）映射成数据库中的记录。每个MyBatis应用程序主要都是使用SqlSessionFactory实例的，一个SqlSessionFactory实例可以通过SqlSessionFactoryBuilder获得。SqlSessionFactoryBuilder可以从一个xml配置文件或者一个预定义的配置类的实例获得。

使用Mybaties的工作流程是首先在配置文件SqlMapConfig.xml中配置数据源、事务等运行环境，以及sql映射文件Map.xml的位置，然后是将SqlMapConfig.xml 配置输入SqlSessionFactory，SqlSessionFactory将会根据配置创建一个会话对象SqlSession，SqlSession向开发者提供增删改查接口，执行增删改查时由SqlSession的内部执行器Executor执行相关操作，其中Map.xml文件配置开发者想要操作的sql语句和方法。

## 2.2 存储相关

### 2.2.1 MySQL数据库

MySQL 是最流行的关系型数据库管理系统，在WEB应用方面 MySQL 是最好的RDBMS(Relational Database Management System：关系数据库管理系统)应用软件之一。

### 2.2.2 Mongodb数据库

MongoDB是当前最流行的Nosql数据库之一。 MongoDB 是由C++语言编写的，是一个基于分布式文件存储的开源数据库系统。在高负载的情况下，添加更多的节点，可以保证服务器性能。MongoDB 旨在为WEB应用提供可扩展的高性能数据存储解决方案。MongoDB 将数据存储为一个文档，数据结构由键值(key=>value)对组成。MongoDB 文档类似于 JSON 对象。字段值可以包含其他文档，数组及文档数组。

### 2.2.3 Ehcache缓存

Ehcache是一个比较成熟的java缓存框架，它提供了用内存，磁盘文件存储，以及分布式存储方式等多种灵活的cache管理方案。EhCache具有如下特点：

（1） 快速简单，非常容易和应用集成。

（2）支持多种缓存策略 。

（3）缓存数据有两级：内存和磁盘，因此无需担心容量问题 。

（4）缓存数据会在虚拟机重启的过程中写入磁盘 。

（5）可以通过RMI、可插入API等方式进行分布式缓存。

（6）具有缓存和缓存管理器的侦听接口 。

（7）支持多缓存管理器实例，以及一个实例的多个缓存区域 等特点。

Ehcache的结构设计概览如下

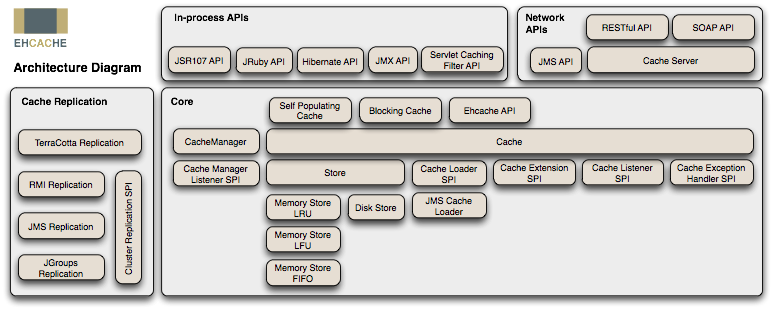


图 3 Ehcache结构

## 2.3爬虫技术

爬虫使用webmagic框架。

WebMagic是一个简单灵活的Java爬虫框架。WebMagic的结构分为Downloader、PageProcessor、Scheduler、Pipeline四大组件，并由Spider将它们彼此组织起来。这四大组件对应爬虫生命周期中的下载、处理、管理和持久化等功能

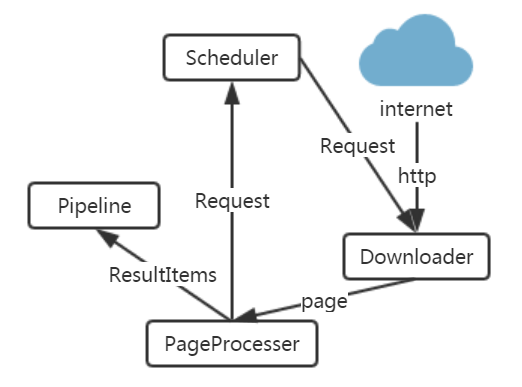


图 4 WebMagic架构

## 2.4 相关框架

### 2.4.1任务调度框架Quartz

Quartz是OpenSymphony开源组织在Job scheduling领域又一个开源项目，是完全由java开发的一个开源的任务日程管理系统，“任务进度管理器”就是一个在预先确定（被纳入日程）的时间到达时，负责执行（或者通知）其他软件组件的系统。

Quartz由Scheduler（任务调度器）、Trigger（触发器）、Calendar（一些日历特定时间点的集合）、JobDetail（描述Job实现类及其它相关的静态信息）、Job（开发者实现该接口定义运行任务）五个核心元素构成。Quartz 支持丰富多样的调度方法，可以满足各种常规及特殊需求；支持任务和调度的多种组合方式，支持调度数据的多种存储方式等优点。Quartz 很容易与 Spring 集成实现灵活可配置的调度功能。



图 5 Quartz核心类关系图

### 2.4.2 消息队列框架ActiveMQ

ActiveMQ是一种开源的，实现了JMS1.1规范的，面向消息(MOM)的中间件，为应用程序提供高效的、可扩展的、稳定的和安全的企业级消息通信。ActiveMQ使用Apache提供的授权，任何人都可以对其实现代码进行修改。

ActiveMQ 支持两种截然不同的消息传送模型：PTP（即点对点模型）和Pub/Sub（即发布 /订阅模型），分别称作：PTP Domain 和Pub/Sub Domain。



图 6 ActiveMQ接收和发送消息基本流程

## 2.5 前端技术

1.AngularJS介绍

AngularJS 是一款来自 Google 的前端 JavaScript 框架，也是 SPA（single-page-application，单页应用）框架。AngularJS 框架的体积非常小，但是设计理念和功能却非常强大，极大地简化前端开发的负担，它快速成为了 JavaScript 的主流框架，帮助开发者从事 web 开发。

AngularJS有五个主要核心特性如下：

①双向数据绑定：实现了把model与view完全绑定在一起，model变化，view也变化，反之亦然。

②模板：在AngularJS中，模板相当于HTML文件被浏览器解析到DOM中，AngularJS遍历这些DOM，也就是说AuguarJS把模板当做DOM来操作，去生成一些指令来完成对view的数据绑定。

③MVVM：吸收了传统的MVC设计模式针但又并不执行传统意义上的MVC，更接近于MVVM(Moodel-View-ViewModel)。

④依赖注入：AngularJS拥有内建的依赖注入子系统，可以帮助开发人员更容易的开发，理解和测试应用。

⑤指令：可以用来创建自定义的标签，也可以用来装饰元素或者操作DOM属性。

## 2.6 本章小结

本章旨在介绍开发过程中使用的相关技术与框架。本章介绍了后端使用的SSM框架、Quartz、WebMagic、ActiveMQ以及前端使用的AngularJs、Angular Material 技术等。

# 3. 推荐算法研究

个性化新闻推荐系统的核心是推荐引擎。现阶段推荐算法已经相当成熟，本结主要是对比较流行的协同过滤与SVD推荐算法研究介绍。

## 3.1 基于协同过滤(Collaborative Filtering)的推荐

协同过滤推荐算法是诞生最早，并且较为著名的推荐算法。主要的功能是预测和推荐。协同过滤可简单的分为两类，分别是基于用户的协同过滤算法(user-based collaboratIve filtering)，和基于物品的协同过滤算法(item-based collaborative filtering)。

### 3.1.1 基于用户的协同过滤算法（userCF）

基于用户的协同过滤算法根据某一用户偏好找到与该用户偏好相似的用户集，然后预测该用户相对相似用户集的未评分项的偏好值，最后根据定义的过滤规则推荐给该用户相关未评分项。

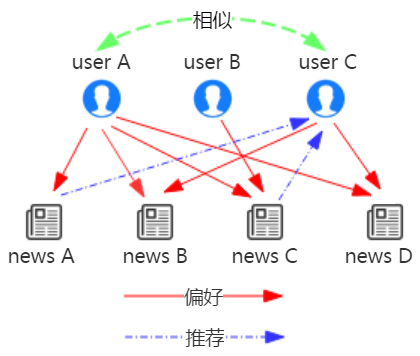


图 7 basic-userCF原理

以个性化新闻推荐举例说明：

设用户新闻偏好矩阵（user-item）为。众所周知在计算机的世界只有数字，因此用户对某一新闻的偏好只能用数字表示，数字的大小表示了该用户对该新闻的喜好程度。如下表所示

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| News  Score  User | News1 | News2 | News3 | News4 | News5 | …… |  |  |
| User1 | 0 | 1 | 2 | 2 | 5 | …… | 0 | 5 |
| User2 | 1 | 1 | 1 | 0 | 5 | …… | 4 | 3 |
| User3 | 2 | 3 | 3 | 3 | 3 | …… | 4 | 4 |
| ……… | …… | …… | …… | …… | …… | …… | …… | …… |
|  | 2 | 2 | 2 | 0 | 0 | …… | 0 | 2 |
|  | 3 | 2 | 1 | 0 | 0 | …… | 5 | 5 |

表 1 user-item举例

上表中用户对于某一条新闻的评分可根据用户相应的历史行为如点击、阅读、收藏、评论等给出，开发者应该合理的制定打分规则，以便真实表达出用户对该新闻的偏好程度。表中0表示该用户对该新闻无任何正反馈行为，即不知道用户对该新闻的偏好程度，默认对某一条新闻无任何反馈行为的为0，且不能为负值；

基于用户的协同过滤算法的核心是找出相似用户。计算两个用户的相似度的方法有多种比如欧几里得距离（Eucledian Distance）、余弦相似度（Cosine Similarity）、调整余弦相似度(Adjusted Cosine Similarity)、皮尔森相关系数(Pearson Correlation Coefficient)、K-近邻算法(K-neighborhood)、曼哈顿距离（Manhattan Distance）、明可夫斯基距离（Minkowski distance）、杰卡德相似性（Jaccard Similarity）等方式。这里我们不对相似度计算算法做过多赘述，我们选取调整余弦相似度(Adjusted Cosine Similarity)作为举例说明。调整余弦相似度(Adjusted Cosine Similarity)公式如下：

公式 1调整余弦相似度(Adjusted Cosine Similarity)

调整余弦相似度(Adjusted Cosine Similarity)是由余弦相似度(Adjusted Cosine Similarity)变化而来。计算时U表示用户的评分集即, ,;

表示用户u对的评分，表示用户u评分集的平均值，同理。

计算出用户之间的相似度后，我们得到了相似度由高到低的N个用户集，我们选取其中前K个用户作为预推荐相似用户即邻居。

最后推荐时我们是从已选出的K个相似用户集中找到推荐用户为阅读的N条新闻推荐给用户。由于K个相似用户中可能或多于或少于N条推荐新闻所以我们需要预测评分。以上面使用的调整余弦相似度(Adjusted Cosine Similarity)为前提，预测评分有加权和回归两种，这里采用加权平均法举例，公式如下：

公式 2加权平均

表示用户u对未评分的的预测评分，表示用户的相似度，表示用户对的实际评分值。

由上面的举例知协同过滤算法的一般步骤如下：



图 8协同过滤算法流程

### 3.1.2 基于项的协同过滤算法（itemCF）

基于项的协同过滤算法是根据用户喜欢的物品与用户未知的物品的相似度，推荐给与用户喜欢的物品最相似的物品。

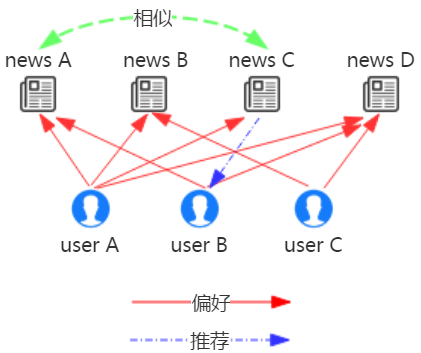


图 9 basic-itemCF原理

基于项的协同过滤算法的计算与基于用户的协同过滤算法大致相同。举例说明如下：

输入数据与上面基于用户的协同过滤算法相同，计算相似度时以上面的调整余弦相似度(Adjusted Cosine Similarity)为例

计算时表示新闻项，U表示新闻的评分集即，,；表示用户对新闻的评分，表示新闻的平均评分，同理。

预测评分时同理可采用回归或加权，但是计算向量都必须是新闻向量。

## 3.2基于SVD的推荐算法

奇异值分解(Singular Value Decomposition，简称SVD)是在机器学习领域广泛应用的算法，它不仅可以用于降维算法中的特征分解，还可以用于推荐系统，以及自然语言处理等领域。

### 3.2.1特征值

定义设**A**是n阶方阵，如存在数λ和n维非零向量**X，**使关系式

公式 3特征方程

成立，则称数λ为方阵**A**的**特征值；**非零向量**X**称为**A**的对应于特征值**λ**的**特征向量**

解特征值与特征向量可以使用公式构建特征方程组。

定理 n阶矩阵**A**与对角矩阵相似的充分必要条件是**A**有n个线性无关的特征向量

当n阶矩阵**A**有n个不同特征值时**A**一定可相似对角化即 **=>**

由上知如果我们求出n阶矩阵**A**的n个特征值以及n个特征值所对应的特征向量，那么矩阵**A**可表示如下

其中**W**是n个特征向量所组成的阶矩阵，**Σ**是n个特征值为对角线的阶矩阵。当**W**为正交矩阵时即**，**因此。

### 3.2.2 SVD

从上面的特征值知我们可以将一个n阶矩阵**A**分解为3个矩阵相乘的形式。但对于一个阶矩阵特征值的方式却不行，因此就有了SVD。SVD是对特征值分解的扩展，它适用于任意阶矩阵。

SVD公式如下

公式 4 SVD

其中都是正交矩阵。

求解三个矩阵步骤如下

1. 求出，；
2. 计算的m个特征值与m个单位正交特征向量，得到由所有特征向量组成的的左奇异向量。
3. 计算的n个特征值与n个单位正交特征向量，得到由所有特征向量组成的矩阵的转置矩阵，的右奇异向量。
4. 计算奇异值σ，因为所以即所以

（注特征值由大到小一一对应；可以使用公式）

计算举例如下：

设矩阵则

；

的特征值与特征向量为

；

的特征值与特征向量为；

求奇异值利用

得奇异值分解如下

### 3.2.3奇异值分解SVD推荐算法

SVD中前10%甚至1%的奇异值的和就占了全部的奇异值之和的99%以上的比例，因此采用近似的描述原矩阵即如下公式：

SVD推荐算法的基本原理是我们计算出奇异值然后取k个奇异值在与左右奇异向量相乘最后得到与原矩阵相似的矩阵，得到的相似矩阵会出现原矩阵未评分的项出现了打分，它可以看成是预测出的，为了使这个预测的分数接近真实值我们需要将相似矩阵与原矩阵的已评分项进行比较，使用某种方法使相似矩阵与原矩阵的已评分项的误差减小，当误差减小到一定程度我们默认相似矩阵与原矩阵已无差别因此相似矩阵中出现的预测评分被认为是用户的真实评分，最后根据这些预测评分推荐给用户合理的项。

SVD推荐算法的核心是预测。SVD推荐的衍生算法有Basic SVD、RSVD、ASVD、SVDPP（SVD++），这些算法都旨在解决预测中的各种问题，预测中问题解决这里不做赘述。

## 3.3奇异值分解SVD与协调过滤的结合

协同过滤中使用奇异值分解主要是为了降维。在实际的推荐系统中用户评分集是一个海量数据，因此使用SVD可有效的减小数据量。

通过奇异分解SVD将矩阵降维成矩阵。将原始数据进行这种降维可行，是因为乘以正交矩阵相当于在矩阵所在的空间进行了旋转缩放等变换，从而转换成为所在的空间，因此不改变矩阵数据的原意。

协同过滤使用SVD，是在输入DataModel层，剩下步骤与协同过滤算法相同。

## 3.4本章小结

本章主要介绍了协同过滤算法与奇异值分解SVD算法，通过用通俗的语言，举例方式说明了协同过滤算法及SVD算法怎样在推荐系统中使用。

# 4. 需求分析

软件需求分析（Software Reguirement Analysis）是研究用户需求得到的东西，完全理解用户对软件需求的完整功能，确认用户软件功能需求，建立可确认的、可验证的一个基本依据。本章将分析个性化新闻推荐的具体需求。

## 4.1需求概述

需求分析涉及方面很多，主要包括三个方面：

1. 在功能方面，需求包括系统要做什么，相当原系统目标系统需求进行哪些修改，目标用户有哪些，以及不同用户需要通过系统完成何种操作等。
2. 在性能方面，需求包括对于系统执行速度、响应时间、吞吐量和并发等指标的要求。
3. 在运行环境方面，需求包括目标系统对于网络设置、硬件设置、温度和湿度等周围环境的要求，以及操作系统、数据库和浏览器等软件配置的要求

本系统主要是提供用户新闻浏览与阅读，以及推荐新闻给用户。通过收集用户使用本系统的一些行为日志为用户对一条新闻的的偏好自动打分。根据这个基本思路我们进行需求分析。



图 10顶层数据流

## 4.2功能性需求

本文设计的个性化新闻推荐系统是通过爬虫技术定时爬取互联网新闻存储的本地仓库，由推荐系统计算后找出用户感兴趣的新闻推送给用户，用户产生行为时，记录用户的这些行为形成日志，日志反馈给评分系统，由评分系统产生用户偏好模型。



图 11系统数据流

本系统包括爬虫系统、新闻处理、用户模块、日志处理、消息处理、任务调度、评分系统、推荐处理功能。



图 12需求模块

### 4.2.1爬虫系统

我们的系统是推荐新闻，我们必须保证系统拥有新闻数据源，因此我们采用爬虫技术爬取网络新闻存储到个人数据仓库作为系统的新闻源。对于新闻网站爬取的网页我们需要过滤html标签，去除重复文本，提取提取出新闻标签生成新闻条目数据。



图 13爬虫数据流

### 4.2.2新闻处理功能

新闻处理包括对新闻，去重，存储，分类功能。在爬虫阶段我们是对爬取的网页处理最终生成新闻条目，爬虫对同一网页的二次爬取可能生相同的新闻条目，因此必须去除重复。对新闻的分类存储有利于新闻信息的管理。



图 14新闻处理数据流

### 4.2.3用户模块功能

用户模块主要包括用户管理功能和用户行为。用户管理是对用户信息的增删改查，对应于系统中的用户登录注册。个性化新闻推荐系统依赖于用户偏好，偏好依赖用户历史行为数据，所以对用户产生或定义的用户历史数据收集是十分重要的。

在本系统中用户可以登录注册本系统，用户可以点击然后阅读新闻，同时可以对新闻进行评论，也可以收藏自己喜欢的新闻，搜索想要的新闻，更新推荐列表。



图 15用户模块功能描述

### 4.2.4日志处理功能

日志处理功能主要是对用户的行为日志进行存储，用户的相关行为包括点击、阅读、收藏、评论、分享。通过注册监听或拦截器实时监控用户行为，将用户的行为日志保存到数据库。日志处理不能影响其他模块，它必须是可分离的。



图 16日志数据流

### 4.2.5消息处理

消息队列的出现主要是解决两个问题

（1）系统解耦：项目开始时，无法确定最终需求，不同进程间，添加一层，实现解耦，方便今后的扩展。

（2）消息缓存：系统中，不同进程处理消息速度不同，MQ，可以实现不同Process之间的缓冲，即，写入MQ的速度可以尽可能地快，而处理消息的速度可以适当调整（或快、或慢）。

本系统使用消息队列，是为了避免爬虫直接操作数据库存储消息。使用消息队列可以减缓存储压力。当爬虫爬取新闻后并不是直接存到数据库，而是发送到消息队列中由消息处理者在空闲时将新闻信息存储到数据库。使用消息队列实现了业务之间的解耦，减少了业务之间的直接依赖。爬虫与新闻处理之间只存在消息的发送与处理不存在业务交叉。



图 17消息队列数据流

### 4.2.6任务调度

任务调度即定时任务，我们对新闻的获取是使用爬虫，我们的爬虫不可能一直处于运行状态，我们希望有新的一条或几条新闻在网络上出现后我们才启动爬虫爬取该新闻以保证系统中新闻的实效性。因此我们必须使用定时调度任务帮助我们在指定时间执行指定任务。



图 18定时任务流程

### 4.2.7评分系统

新闻推荐依赖用户的偏好模型，怎样建立一个合理的偏好模型是非常重要的。本系统采用的是分析用户历史行为日志和用户其他数据，为用户进行新闻评分的形式建立用户偏好模型。

在分析获取用户数据时分为显式和隐式获取。显式是在用户登录系统的前提下直接访问本地日志；隐式是用户未登录系统的前提下主动获取用户浏览器的相关历史信息并结合本地日志进行评分。

### 4.2.8推荐处理

推荐处理是本系统的核心。推荐处理的功能主要是实现个性化推荐算法计算用户偏好模型即DataModel，给出推荐结果。

## 4.3实体-关系图

实体-关系图即E-R图（Entity Relationship Diagram），是指提供了表示实体型、属性和联系的方法，用来描述现实世界的概念模型。E-R方法是“实体-关系方法”（Entity-Relationship Approach）的简称。它是描述现实世界概念结构模型的有效方法。

综合以上需求分析可得本系统E-R图如下



图 19 E-R图

日志存储使用的是NoSQL数据库Mongodb，日志类图如下



图 20日志类

## 4.4非功能性需求

软件需求被分为两大类分别是功能性需求和非功能性需求。其中非功能性需求是常常被轻视，甚至被忽视的一个重要方面。其实，软件产品非功能性定义不仅决定产品的质量，还在很大程度上影响产品的功能需求定义。如果事先缺乏很好的非功能性需求定义，结果往往是使产品在非功能性需求面前捉襟见肘，甚至淹没功能性需求给用户带来的价值。非功能性需求，是指软件产品为满足用户业务需求而必须具有且除功能需求以外的特性其中包括系统性能、可靠性、可维护性，可扩充性以及对技术与业务方面的适应性。

本工程的非功能性需求可在日后完成原型的基础改进。

## 4.5本章小结

本章主要对系统进行了需求分析，在本章中我主要是通过图形和文字说明的方式来介绍本系统的功能需求。

# 5. 系统设计与实现

本系统使用SSM框架结合多种java技术实现，整个系统分为结果显示的前端和逻辑业务处理的后端两部分。由前一章的功能需求分析得到整个系统的功能结构模型如下



图 21功能结构

## 5.1系统架构

整个系统按业务需求可划分为三层结构分别是表示层、业务逻辑层、数据访问层



图 22个性化新闻推荐系统分层结构

1. 表示层：本系统的表示层由web页面负责，主要的功能是显示数据，提供的页面有用户登录注册页、推荐列表页、新闻列表页、热点新闻页、新闻详情与评论页。
2. 业务逻辑层：这是系统的核心，它包含许多功能日志管理、偏好模型建立功能、新闻管理、评论、推荐功能。业务逻辑层主要向表示层传递计算结果，向数据访问层获取数据
3. 数据访问层：数据访问层负责数据的访问。本系统使用MySQL、Mongodb作为数据库，MySQL的访问使用Mybatis，Mongodb的访问使用Spring插件。

## 5.2数据库设计

（1）users表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 键名 | 类型 | 约束 | default | 说明 |
| id | bigint(20) | auto\_increment not null  primary key |  | 用户编号 |
| name | varchar(20) | not null unique |  | 用户名 |
| password | varchar(20) | not null |  | 用户密码 |
| avatar | varchar(20) |  |  | 用户头像 |
| gmt\_create | datetime | not null | CURRENT\_TIMESTAMP | 创建时间 |
| gmt\_modified | datetime | not null | CURRENT\_TIMESTAMP  ON UPDATE CURRENT\_TIMESTAMP | 修改时间 |

表 2 users

（2）news\_catalog表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 键名 | 类型 | 约束 | default | 说明 |
| id | bigint(20) | auto\_increment not null  primary key |  | 类别编号 |
| name | varchar(100) | not null unique |  | 类别名 |
| gmt\_create | datetime | not null | CURRENT\_TIMESTAMP | 创建时间 |
| gmt\_modified | datetime | not null | CURRENT\_TIMESTAMP  ON UPDATE CURRENT\_TIMESTAMP | 修改时间 |

表 3 news\_catalog

（3）news表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 键名 | 类型 | 约束 | default | 说明 |
| id | bigint(20) | auto\_increment not null  primary key |  | 新闻编号 |
| title | varchar(40) | not null unique |  | 新闻标题 |
| url | varchar(255) | not null |  | 新闻URL |
| media | varchar(20) |  |  | 新闻媒体 |
| imgurl | varchar(255) |  |  | 新闻图片URL |
| content | varchar(255) |  |  | 备注 |
| catalog\_id | bigint(20) | not null |  | 类别id |
| gmt\_create | datetime | not null | CURRENT\_TIMESTAMP | 创建时间 |
| gmt\_modified | datetime | not null | CURRENT\_TIMESTAMP  ON UPDATE CURRENT\_TIMESTAMP | 修改时间 |

表 4 news

（4）comment表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 键名 | 类型 | 约束 | default | 说明 |
| id | bigint(20) | auto\_increment not null  primary key |  | 评论编号 |
| user\_id | bigint(20) | foreign key  on delete cascade |  | 用户id |
| news\_id | bigint(20) | foreign key  on delete cascade |  | 新闻id |
| content | varchar(400) |  |  | 评论 |
| gmt\_create | datetime | not null | CURRENT\_TIMESTAMP | 创建时间 |
| gmt\_modified | datetime | not null | CURRENT\_TIMESTAMP  ON UPDATE CURRENT\_TIMESTAMP | 修改时间 |

表 5 comment

（5）news\_preferences表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 键名 | 类型 | 约束 | default | 说明 |
| id | bigint(20) | auto\_increment not null  primary key |  | 编号 |
| user\_id | bigint(20) | foreign key  on delete cascade |  | 用户id |
| news\_id | bigint(20) | foreign key  on delete cascade |  | 新闻id |
| preference | integer | not null | 0 | 偏好程度 |
| gmt\_create | datetime | not null | CURRENT\_TIMESTAMP | 创建时间 |
| gmt\_modified | datetime | not null | CURRENT\_TIMESTAMP  ON UPDATE CURRENT\_TIMESTAMP | 修改时间 |

表 6 news\_preferences

Mongodb中的文档News类文档，BehaviorLog类文档。

## 5.3环境说明

（1）开发环境

①数据库：Mysql-5.7.16、mongodb-3.6.4

②开发平台：eclipse4.5.0

③环境配置：jdk1.8.6、tomcat8、maven3

（2）运行环境

CPU：intel i5以上

内存：4GB以上

硬盘空间：2GB以上

网络速度：10M以上

操作系统：window7/linux

## 5.4实现说明

（1）用户模块

用户模块是与用户相关的功能模块，比如登录注册这些都实现在UserService中。

（2）爬虫模块

爬虫技术是一门复杂的技术，这里主要使用的是爬虫框架WebMagic。使用Webmagic时，运行入口需要实现接口PageProcessor。爬虫是由定时器启动的，在Spring中需要配置Quartz，xml设置的bean有JobDetailFactoryBean、CronTriggerFactoryBean、SchedulerFactoryBean，爬虫爬取的新闻发送到ActiveMQ消息队列，最后的存储由NewsService类完成。

（3）评分系统

对于用户偏好评分时我们使用公式

对于用户行为权重我们这里依从简单设置默认权重都为1

（4）评论系统

用户对于新闻可评论应该是用户已登录本系统，设置评论系统是为了以后用户偏好模型建立扩展时可以有更多的用户信息，可以使用评分分析用户对该新闻的情感。

（5）推荐模块

这里推荐使用Mahout算法框架，Mahout包含许多推荐算法。数据输入使用NewsDataModel继承MySQLJDBCDataModel输入news\_preferences表的数据。

（6）日志

行为日志的实现交由BehaviorLogService类，基本原理是监听API请求因此我们需要实现一个继承HandlerInterceptorAdapter的拦截类并且在Spring中使用<mvc:interceptors>注册拦截器。

（7）前端

整个前端使用angularjs路由进行单页面开发，整个前端应用为app模块，其余业务逻辑封装在MyService模块中，Myservice模块中使用$q服务异步请求后端数据，页面组件我们使用的是Angular Material组件。

对于用户没有登录注册的请求推荐新闻情况，个性化推荐是针对个人的因此必须要有唯一身份标志一个用户考虑使用ip或session都不可行，我们知道用户对于个性化推荐是由黏性的，因此我们假定使用同一地点浏览器浏览本系统的用户是不变的即用户的个人浏览器不会有第二个人使用，我们将未登录注册的用户历史行为及偏好记录在本地每次请求推荐时都提交用户本地的对本系统的历史行为日志及本地记录的个人偏好模型，作为推荐计算依据，推荐身份我们始终分配用户id为0且系统后台不存储该用户的偏好模型。本地记录使用localstorage存储，前端日志依然使用监听拦截请求的方式。Angularjs拦截是在config中配置 $httpProvider.interceptors.push(‘拦截类’)，拦截类实现request、requestError、response、responseError四个方法可以拦截请求时，请求出错时，后台响应成功时，后台出错时。

# 6. 系统测试

软件测试是系统开发必不可少的，通过测试可以及早发现系统的相关错误和不足，及早的修复系统的相关缺陷。

本系统开发使用Junit4+spring框架测试。JUnit是一个单元测试框架，它使用注解的形式来指定需要测试的方法。JUnit测试其实就是执行一个类中的方法，并且这个类仅用于测试，称之为测试类（Test Class），这个方法使用 @Test 注解标注。这个方法执行待测试的代码，可以通过使用JUnit或者其他的Assert（断言）框架提供的Assert相关的方法来检查预期结果和实际结果是否相符。这些方法的调用通常称之为断言（asserts ）或者断言语句（assert statements）。Spring test framework是spring的测试框架，Spring test framework提供 @ContextConfiguration 和 @Configuration 两个注解，使用这俩个注解，可以使测试类自动加载 XML 文件，获得application context。

前端测试主要是对请求测试，测试使用的是Mock.js请求拦截模拟框架。

# 7. 总结与展望

通这次的毕业设计我了解了协同过滤算法和SVD算法，协同过滤是以计算相似度的方式实现个性化推进，SVD是以回归预测的方式实现个性化推荐，这两个基本推荐算法实现起来并难，但是一个真正可应用的推荐算法需要考虑诸多因素还有与时俱进不断完善，比如SVD，SVD推荐算法就是奇异值分解然后取近似奇异值在乘回去得到相似矩阵然后使用梯度下降法减小误差最后推荐，听起来简单，但是实际推荐中又会出现过拟合问题，因此又出现了RSVD（正则化SVD），之后好像没有什么问题了，但我们又发现这个算法还可以优化一下所以出现了对偶算法，现在差不多优化完了我们应用到实际中，但是数据量很大，于是思考要是有大数据框架，可是数据顺序计算不够高效，那么思考并行运算，怎样结合并行运算，所以出现了以Spark框架基础的推荐算法。

对于我所实现的个性化新闻推荐系统只是一个原型真正运行使用还有一段距离，设计实现这个系统除了研究推荐算法外另一个目的就是软件工程的思维与编程经验。一个软件诞生的背后有许多事情是需要做，这就是软件工程的定义。我们开发一个软件就有思考为什么要花时间开发它；它是做什么的；它的技术难点是什么；它是可完成的吗等等问题，在确定开发后就要思考怎么做于是就要进行需求分析、概要设计之类。等到梳理完这些需求功能后就要代码实现，这时候环境搭建，运行测试不定时就会有个异常出现，那我们就必须查资料并思考解决这个问题。经历整个过程我们就有了软件工程思维，提高了编程技巧。一个软件开发完了，只有源代码是不够的，我们必须提供文档。编写文档也不是一个简单的复制粘贴的过程。思考合理的表达，或提供独有的见解，怎样的组织文档，怎样概括等都是编写文档必须考虑的，这也是一种锻炼。

本毕业设计更合理的应该使用大数据框架实现，但本人基于自我经验选择了基本的SSM框架，因此本系统可以在以后扩展到大数据框架。在算法方面，人工智能深度学习是一个热点，我对深度学习的理解就是给出输入输出深度学习就是给出中间执行的过程方法即一个计算输入得到输出的过程，深度学习与大数据有良好的相吸性，因此使用深度学习做个性化推荐算法可以是今后改进本系统个性化新闻推荐算法的一个方向。

由于各种原因如有限的知识水平和匮乏的经验以及有限的时间，导致此次软件设计和实现中无可避免存在很多瑕疵，这些都是成长的道路路上必需经历的。对我来说这次毕业设计的软件开发是一个不小的挑战，通过实践的锻炼，让我明白了学习的重要性，并从中积累了不少解决问题的方法和思路，为以后的工作打下了良好的基础。

# 参考文献

[1]推荐系统综述 作者张韩飞

[2]《个性化推荐系列之推荐系统否认演化及常见推荐算法》

[3]孟婷婷. 基于社交网络的推荐算法应用研究[D]. 重庆大学, 2015.

[4]熊李媛 个性化新闻推荐系统的研究与设计 20170325

[5]李勇，徐振宁．Intemet个性化信息服务研究综述[J]，计算机工程与应用，2002，19；183-187．

[6]基于改进SVD算法的个性化新闻推荐系统设计和实现 杨鹏 计算机技术 西安电子科技大学 2015（学位年度）

[7]基于 Android 智能终端的个性化新闻推荐系统 李洁 周晨程 陈慧萍 华娇娇 赵絗絗 秦莹莹 《微处理机》 CSTPCD - 2015年3期

[8]基于内容的个性化新闻推荐 蒲国林 王刚 向伟 杨勇智 《四川文理学院学报》 - 2013年5期

[9]陈清浩. 基于SVD的协同过滤推荐算法研究[D]. 西南交通大学,2015:1-72.

[10]基于SVD算法的智能新闻推荐系统的设计与实现 洪英汉，郭才（韩山师范学院，广东 潮州 521041)

[11] 刘 建 国 , 周 涛 , 汪秉宏等 . 个性化推荐系统的研究进展 [J]. 自 然 科 学 进展,2009,19(1):1-15.DOI:10.3321/j.issn:1002-008X.2009.01.001.

[12] 李忠俊,周启海,帅青红等.一种基于内容和协同过滤同构化整合的推荐系统模型[J].计算机科学,2009,36(12):142-145.DOI:10.3969/j.issn.1002-137X.2009.12.034.

[13] 商雪晶.基于内容的相关书籍推荐技术研究[D].哈尔滨工业大学,2010.

[14] 田久乐,赵蔚.基于同义词词林的词语相似度计算方法[J].吉林大学学报（信息学

版）,2010,28(6):602-608.DOI:10.3969/j.issn.1671-5896.2010.06.011.

[15] 郭艳红 . 推 荐 系 统 的 协 同 过 滤 算 法 与 应 用 研 究 [D]. 大 连 理 工 大

学,2008.DOI:10.7666/d.y1419326.

[16] 杜乐然.电子商务推荐系统的设计与实现[D].大连理工大学,2013.

[17] 罗 荣 , 唐学兵 . 基 于 JDBC 的 数 据 库 连 接 池 的 设 计 与 实 现 [J]. 计算机工程,2004,30(9):92-93,111.DOI:10.3969/j.issn.1000-3428.2004.09.037.

[18]M. Pazzani, D. Billsus, Content-based recommendation systems, in: P.Brusilovsky, A. Kobsa, W. Nejdl (Eds.) The Adaptive Web, Springer BerlinHeidelberg2007, pp. 325-341.

[19]Salton G. Automatic text processing[M]. Addison-Wesley LongmanPublishing Co. Inc. 1989.

[20]Deerwester S. Indexing by latent semantic analysis[J]. Journal ofthe Association for Information Science & Technology, 1990, 41(6):391-407.

[21]Hofmann T. Unsupervised Learning by Probabilistic Latent SemanticAnalysis[J]. Machine Learning, 2001, 42(1-2):177-196.

[22]M. Deshpande, G. Karypis, Item-based top-N recommendationalgorithms, ACM Transactions on Information Systems (TOIS), 22 (2004) 143-177

[23]P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl,GroupLens: an open architecture for collaborative filtering of netnews,  Proceedings of the 1994 ACM Conference onComputer Supported Cooperative Work, ACM, Chapel Hill, North Carolina, USA,1994, pp. 175-186.

[24]Lu J, Wu D, Mao M, et al.Recommender system application developments[J]. Decision Support Systems, 2015,74(C):12-32.

[25]M. Nilashi, O.b. Ibrahim, N.Ithnin, Multi-criteria collaborative filtering with high accuracy using higherorder singular value decomposition and Neuro-Fuzzy system, Knowledge-BasedSystems, 60 (2014) 82-101.

[26]G.-R. Xue, C. Lin, Q. Yang, W.Xi, H.-J. Zeng, Y. Yu, Z. Chen, Scalable collaborative filtering usingcluster-based smoothing,  Proceedings ofthe 28th Annual International ACM SIGIR Conference on Research and Developmentin Information Retrieval, ACM, Salvador, Brazil, 2005, pp. 114-121.

[27]S.K. Shinde, U. Kulkarni,Hybrid personalized recommender system using centering-bunching basedclustering algorithm, Expert Systems with Applications, 39 (2012) 1381-1387.

# 图表公式目录

图目录

[图 1 Spring 框架总览 11](#_Toc515990975)

[图 2 SpingMVC框架 12](#_Toc515990976)

[图 3 Ehcache结构 14](#_Toc515990977)

[图 4 WebMagic架构 14](#_Toc515990978)

[图 5 Quartz核心类关系图 15](#_Toc515990979)

[图 6 ActiveMQ接收和发送消息基本流程 16](#_Toc515990980)

[图 7 basic-userCF原理 18](#_Toc515990981)

[图 8协同过滤算法流程 20](#_Toc515990982)

[图 9 basic-itemCF原理 21](#_Toc515990983)

[图 10顶层数据流 25](#_Toc515990984)

[图 11系统数据流 26](#_Toc515990985)

[图 12需求模块 26](#_Toc515990986)

[图 13爬虫数据流 27](#_Toc515990987)

[图 14新闻处理数据流 27](#_Toc515990988)

[图 15用户模块功能描述 27](#_Toc515990989)

[图 16日志数据流 28](#_Toc515990990)

[图 17消息队列数据流 28](#_Toc515990991)

[图 18定时任务流程 28](#_Toc515990992)

[图 19 E-R图 30](#_Toc515990993)

[图 20日志类 31](#_Toc515990994)

[图 21功能结构 32](#_Toc515990995)

[图 22个性化新闻推荐系统分层结构 33](#_Toc515990996)

表目录

[表 1 user-item举例 19](#_Toc515845589)

[表 2 users 34](#_Toc515845590)

[表 3 news\_catalog 34](#_Toc515845591)

[表 4 news 34](#_Toc515845592)

[表 5 comment 35](#_Toc515845593)

[表 6 news\_preferences 35](#_Toc515845594)

公式

[公式 1调整余弦相似度(Adjusted Cosine Similarity) 19](#_Toc515845649)

[公式 2加权平均 19](#_Toc515845650)

[公式 3特征方程 21](#_Toc515845651)

[公式 4 SVD 22](#_Toc515845652)

# 源码附录

源代码已上传github，地址：https://github.com/MaloneRe/graduate.git