

**实践成果三—报告**

**专 业： 软件工程\_\_\_\_\_\_\_**

**课题名称：基于协同过滤算法的论文推荐系统**

**班 级： 卓越171\_\_\_\_\_\_\_**

**学 号： 201719244324\_\_\_ \_**

**学生姓名： 杨松桦 \_\_\_\_\_\_\_**

**指导教师： 贾晓辉\_\_\_\_\_\_\_\_**

**2019年 6 月 20日**

目录

[第一章 绪论 2](#_Toc12016171)

[1.1 推荐系统概述 2](#_Toc12016172)

[1.2 推荐系统的背景 3](#_Toc12016173)

[1.3推荐系统的应用 3](#_Toc12016174)

[1.3.1 电子商务网站 4](#_Toc12016175)

[1.2.2 电影和视频网站 5](#_Toc12016176)

[1.2.3 个性化音乐网络电台 6](#_Toc12016177)

[1.2.4 社交网站 7](#_Toc12016178)

[1.2.5个性化阅读 8](#_Toc12016179)

[1.2.6 其他推荐的应用 9](#_Toc12016180)

[1.3 推荐系统的相关技术 10](#_Toc12016181)

[1.3.1 基于内容推荐 10](#_Toc12016182)

[1.3.2协同过滤推荐 10](#_Toc12016183)

[1.3.3 基于关联规则推荐 11](#_Toc12016184)

[1.3.4 基于效用推荐 11](#_Toc12016185)

[1.3.5 基于知识推荐 11](#_Toc12016186)

[1.3.6 组合推荐 12](#_Toc12016187)

[1.3.7 基于深度学习的推荐技术 12](#_Toc12016188)

[1.4 实践成果背景 13](#_Toc12016189)

[第二章 需求分析 15](#_Toc12016190)

[2.1 业务分析 15](#_Toc12016191)

[2.2 系统功能需求分析 15](#_Toc12016192)

[2.3 系统性能需求分析 16](#_Toc12016193)

[2.4系统数据需求分析 16](#_Toc12016194)

[第三章 系统设计 17](#_Toc12016195)

[3.1 系统功能设计 17](#_Toc12016196)

[3.2算法设计 19](#_Toc12016197)

[3.2.1 用户行为分析 19](#_Toc12016198)

[3.2.2 协同过滤算法 20](#_Toc12016199)

[3.2.3 两种协同过滤算法对比 21](#_Toc12016200)

[3.2.4 基于用户的协同过滤算法 21](#_Toc12016201)

[3.2.5 基于物品的协同过滤算法 22](#_Toc12016202)

[3.4 数据库设计 25](#_Toc12016203)

[3.4.1 数据库的详细设计 25](#_Toc12016204)

[第四章 系统实现 28](#_Toc12016205)

[4.1 系统界面实现 28](#_Toc12016206)

[4.1.1 登录/注册界面 28](#_Toc12016207)

[4.1.2 主页面 29](#_Toc12016208)

[4.1.3 搜索/分类显示页面 30](#_Toc12016209)

[4.1.4 论文阅读页面 31](#_Toc12016210)

[4.2 功能模块实现 32](#_Toc12016211)

[4.2.1 注册\登陆模块 32](#_Toc12016212)

[4.2.2搜索功能 34](#_Toc12016213)

[4.2.3推荐模块 35](#_Toc12016214)

[4.2.3 主页模块实现 37](#_Toc12016215)

[第五章 总结 39](#_Toc12016216)

[参考文献： 40](#_Toc12016217)

第一章 绪论

# 1.1 推荐系统概述

自从1950年互联网这一高新技术诞生以来, 在快速推荐了社会产业融合和人类发展的同时, 也一直呈递出一种指数型的发展趋势. 而存储在互联网中大量的数据的增长长度导致想要表示的可计量单位都在迅速变化.

近年来，伴随着云计算、-大数据、物联网、人工智能等信息技术的快速发展和传统产业数字化的转型，数据量呈现几何级增长，据IDC预测，全球数据总量预计2020年达到44个ZB，我国数据量将达到8060个EB，占全球数据总量的18%。

时至今日互联网每天新增的数据量达2.5\*10^18字节，而全球90%的数据都是在过去的两年间创造出来的。举个直观的例子来说明一下互联网的数据量：假设大西洋里每一升海水代表一个字节的数据，那么整个大西洋存储的数据也只能到2010年就满了。从外行的角度看来大数据是个挺了不起的东西，它也确实了不起，不过有一个前提就是我们能够有效地处理数据。

在这种飞速迭代的发展情况之下, 前两年我们还为这互联网为我带带来的信息遍历, 如今又迭代为了一个极为严重的问题 – 信息过载. 随着信息技术和互联网的发展，人们逐渐从信息匮乏的时代走入了信息过载（information overload）的时代。在这个时代，无论是信息消费者还是信息生产者都遇到了很大的挑战：作为信息消费者，如何从大量信息中找到自己感兴趣的信息是一件非常困难的事情；作为信息生产者， 如何让自己生产的信息脱颖而出，受到广大用户的关注，也是一件非常困难的事情。推荐系统就 是解决这一矛盾的重要工具。推荐系统的任务就是联系用户和信息，一方面帮助用户发现对自己 有价值的信息，另一方面让信息能够展现在对它感兴趣的用户面前，从而实现信息消费者和信息 生产者的双赢.

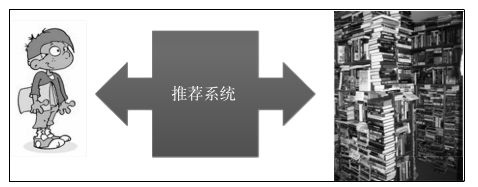


图 1.1 推荐系统

# 1.2 推荐系统的背景

如今，我们这代人正经历从**信息时代**(Information Technology , IT)到**数据时代**(Data Technology , DT)的变迁，DT时代比较明显的标志就是：**信息过载**。

在DT时代，充斥着海量的信息，如何从海量的信息中快捷的帮助特定用户找到感兴趣的信息呢？有两种相关的解决技术：**搜索引擎与推荐系统。**

如果用户有着明确的需求, 那么他完全可以使用搜索引擎. 但是，如果用户没有明确的需求呢？然而大多数适合, 我们其实都没有明确的需求.比如你今天很无聊，想下载一部电影看看。但当你打开某个下载网站，面对100年来发行的数不胜数的电影，你会手足无措，不知道该看哪一部。此时， 你遇到了信息过载的问题，需要一个人或者工具来帮助你做筛选，给出一些建议供你选择。如果 这时候有个喜欢看电影的朋友在身边，你可能会请他推荐几部电影。不过，总不能时时刻刻都去麻烦“专家”给你推荐，你需要的是一个自动化的工具，它可以分析你的历史兴趣，从庞大的电影库中找到几部符合你兴趣的电影供你选择。这个工具就是**个性化推荐系统**.



图 1.2 现状

与搜索引擎不同，推荐系统不需要用户准确地描述出自己的需求，而是根据分析历史行为建模，主动提供满足用户兴趣和需求的信息。从物品的角度出发，推荐系统可以更好地发掘物品的长尾（long tail）

推荐系统是一种信息过滤系统，用于预测用户对物品的“评分”或“偏好”。 推荐系统产生推荐列表的方式通常有两种：[协同过滤](https://zh.wikipedia.org/wiki/%E5%8D%94%E5%90%8C%E9%81%8E%E6%BF%BE)以及[基于内容推荐](https://zh.wikipedia.org/w/index.php?title=%E5%9F%BA%E4%BA%8E%E5%86%85%E5%AE%B9%E6%8E%A8%E8%8D%90&action=edit&redlink=1)，或者基于个性化推荐。 [协同过滤](https://zh.wikipedia.org/wiki/%E5%8D%94%E5%90%8C%E9%81%8E%E6%BF%BE)方法根据用户历史行为（例如其购买的、选择的、评价过的物品等）结合其他用户的相似决策建立模型。这种模型可用于预测用户对哪些物品可能感兴趣（或用户对物品的感兴趣程度）。[基于内容推荐](https://zh.wikipedia.org/w/index.php?title=%E5%9F%BA%E4%BA%8E%E5%86%85%E5%AE%B9%E6%8E%A8%E8%8D%90&action=edit&redlink=1)利用一些列有关物品的离散特征，推荐出具有类似性质的相似物品。两种方法经常互相结合（参考[混合推荐系统](https://zh.wikipedia.org/w/index.php?title=%E6%B7%B7%E5%90%88%E6%8E%A8%E8%8D%90%E7%B3%BB%E7%BB%9F&action=edit&redlink=1)）

# 1.3推荐系统的应用

由于BT时代的特殊背景和用户的强烈需求, 目前对于推荐系统的应用可以说以及是遍地开完, 只要一个成型成体系的信息化网站, 都面临着巨大的推荐系统需求, 尽管不同的网站使用不同的推荐系统技术，但总地来说，几乎所有的推荐系统应用都是由前台的展示页面、后台的日志系统以及推荐算法系统3部分构成的。

从网站分类上看, 目前主流大幅度应用推荐系统的网站主要有着以下几种:

## 1.3.1 电子商务网站

电子商务网站是个性化推荐系统的一大应用领域。在国外著名的电子商务网站亚马逊是个性化推荐系统的积极应用者和推广者，被RWW（读写网）称为“推荐系统之王”。

而随着近些年中国互联网的飞速网站，也涌现出了一大批优秀的电子商务网站。例如下图中国内最大的电商网站，可以看到网站内包括主页的各个地方和角落充斥着推荐的内容，这背后无疑有着盘根错节的推荐算法。

商品推荐页主要包括的内容有

（1）推荐结果的标题，缩略图以及其他内容属性

（2）推荐结果的评分

（3）推荐理由。

亚马逊的推荐有以下三种：

（1）基于用户之前的行为，例如购买过武侠小说，会继续推荐别的武侠小说。

（2）基于用户的好友关系，例如亚马逊拿到用户的Facebook的好友，然后向用户推荐他的好友都买了啥

（3）基于物品的相似度，例如购买了该商品的用户还买了哪些商品。

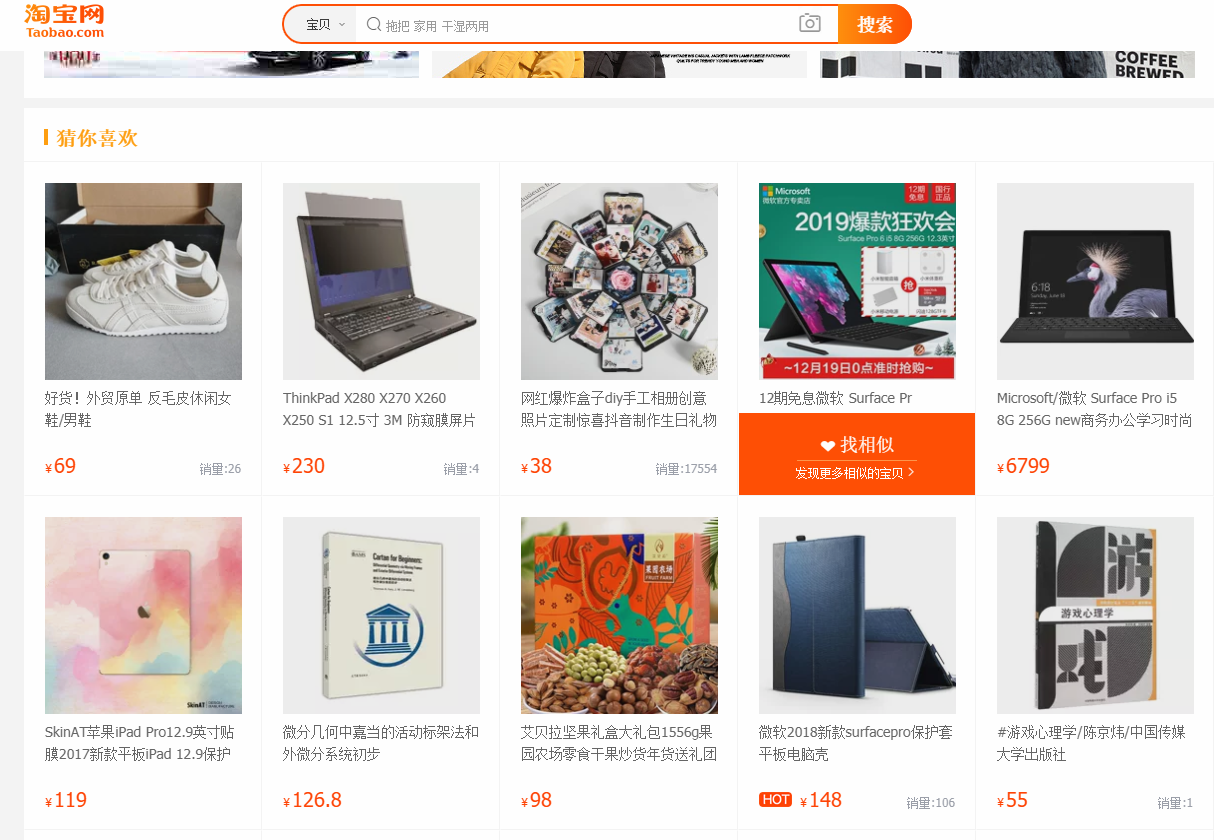


图1.3 淘宝网推荐页面

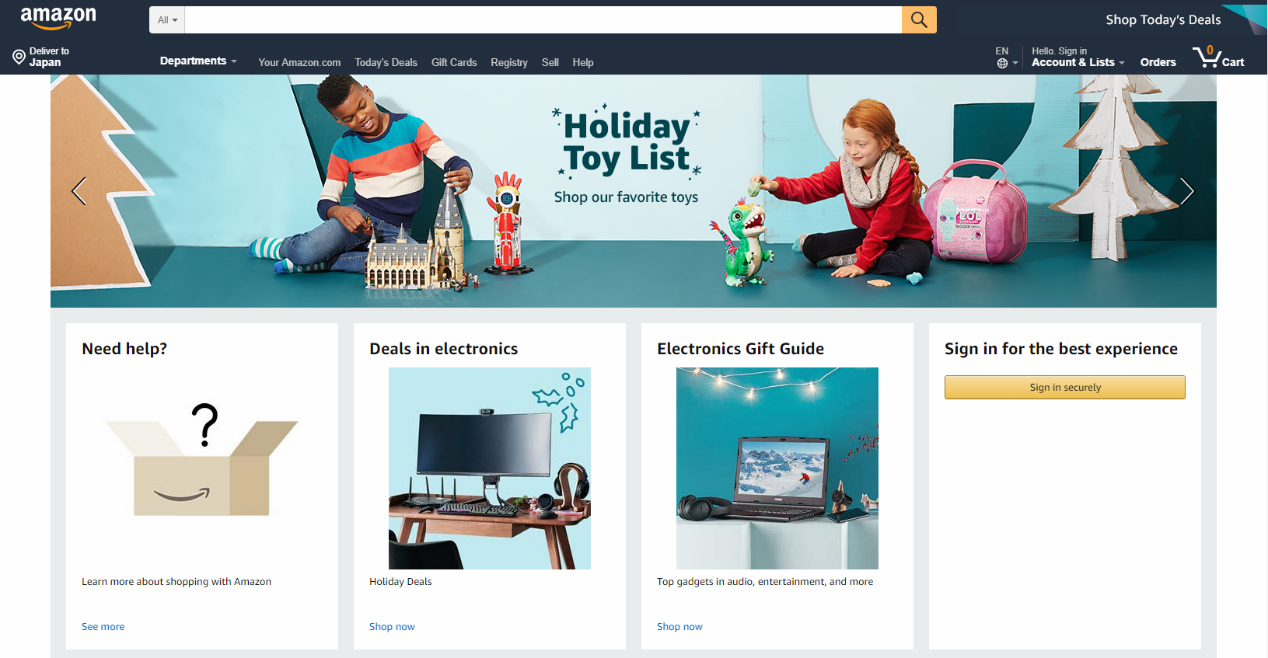


图1.4 亚马逊推荐页面

## 1.2.2 电影和视频网站

在电影和视频网站中，个性化推荐系统也是一种重要的应用。这种和电子商务不太一样，这种用户通常只是想看电影，但是并没有很明确的需求要看那部电影甚至是哪种类型的电影。它能够帮助用户在浩瀚的视频库中找到令他们感兴趣的视频。在该领域成功使用推荐系统的公司就是Netflix和YouTube，其中Netflix和亚马逊是推荐系统领域最具代表性的两家公司。

YouTube作为美国最大的视频网站，拥有大量用户上传的视频内容。由于视频数量非常大，用户在YouTube中面临着严重的信息过载问题。为此，YouTube在个性化推荐领域也进行了深入研究，尝试了很多算法。在YouTube最新的论文中，他们的研究人员表示现在使用的也是基于物品的推荐算法。为了证明个性化推荐的有效性，YouTube曾经做个一个实验，比较了个性化推荐的 点击率和热门视频列表的点击率，实验结果表明个性化推荐的点击率是 热门视频点击率的两倍。



图 1.5 Youtube主页充斥着推荐

## 1.2.3 个性化音乐网络电台

个性化推荐的成功应用需要两个条件。第一是存在信息过载，因为如果用户可以很容易地从所有物品中找到喜欢的物品，就不需要个性化推荐了。第二是用户大部分时候没有特别明确的需 求，因为用户如果有明确的需求，可以直接通过搜索引擎找到感兴趣的物品。

在这两个条件下，个性化网络电台无疑是最合适的个性化推荐产品。首先，音乐很多，用户不可能听完所有的音乐再决定自己喜欢听什么，而且每年新的歌曲在以很快的速度增加，因此用 户无疑面临着信息过载的问题。其次，人们听音乐时，一般都是把音乐作为一种背景乐来听，很 少有人必须听某首特定的歌。对于普通用户来说，听什么歌都可以，只要能够符合他们当时的心 情就可以了。因此，个性化音乐网络电台是非常符合个性化推荐技术的产品。

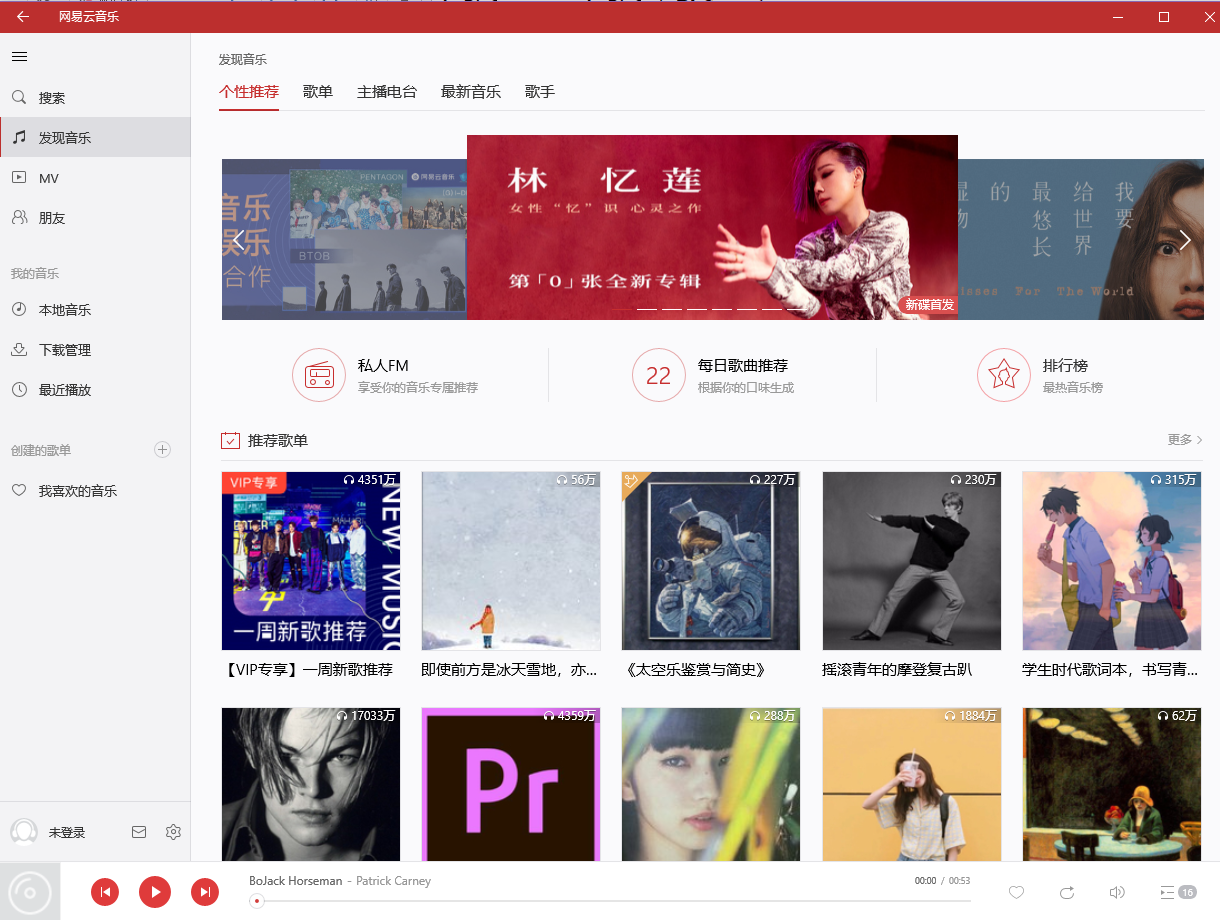


图1.6 网易云音乐主页的推荐

## 1.2.4 社交网站

最近5年，互联网最激动人心的产品莫过于以Facebook和Twitter为代表的社交网络应用。而在国内，目前最成功的社交网站当然是QQ和微信。社交网站最宝贵的数据有两个，一个是用户之间的社交网络关系，另一个是用户的偏好信息。

除了利用用户在社交网站的社交网络信息给用户推荐本站的各种物品，社交网站本身也会利用社交网络给用户推荐其他用户在社交网站的会话。如图1-15所示，每个用户在QQ的查找首页都能看到好友的各种分享，并且能对这些分享进行评论。

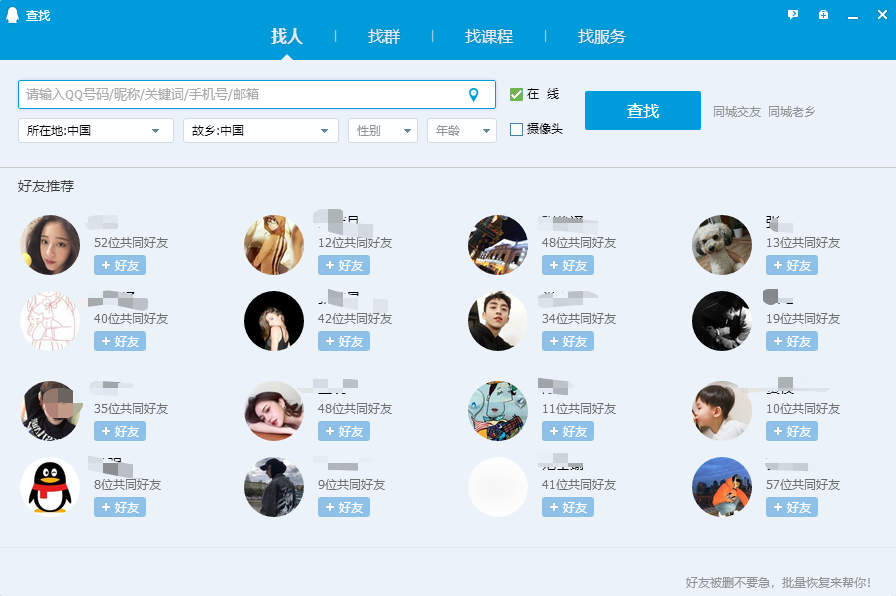


图 1.7 QQ设计界面的推荐

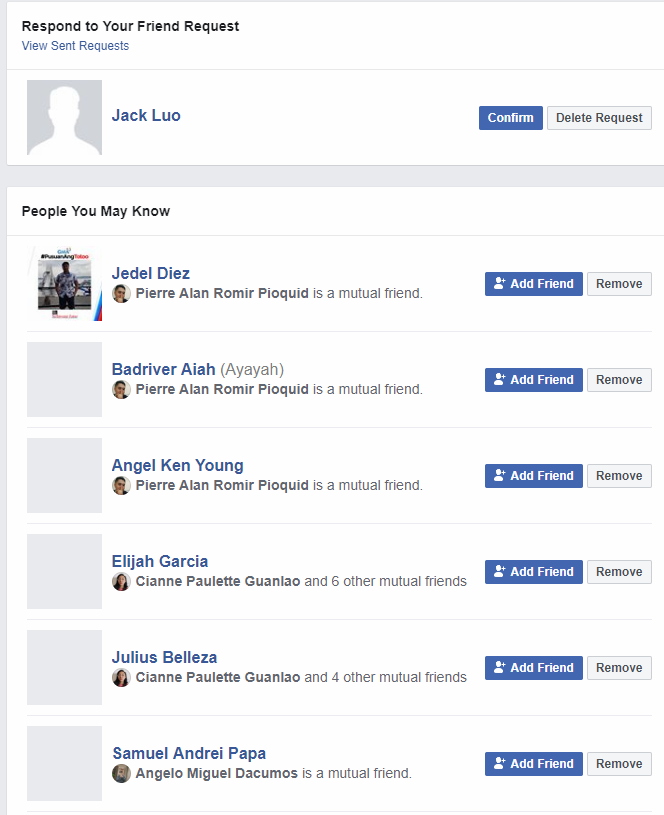


图1.8 Facebook界面的推荐

## 1.2.5个性化阅读

阅读文章是很多互联网用户每天都会做的事情。个性化阅读同样符合前面提出的需要个性化

推荐的两个因素：首先，互联网上的文章非常多，用户面临信息过载的问题；其次，用户很多 时候并没有必须看某篇具体文章的需求，他们只是想通过阅读特定领域的文章了解这些领域的动态。

目前互联网上的个性化阅读工具很多，国际知名的有Google Reader (已经下架)，国内有知乎网等。同时，随着移动设备的流行，移动设备上针对个性化阅读的应用也很多。

知乎网是一款流行的社会化阅读工具，他的文章列表分为推荐、关注、热榜三块，其中推荐是最受欢迎的阅读模块。知乎网可以根据用户的关注领域、以往的点赞、点踩、收藏等有效的针对用户进行个性化推荐，其推荐算法收到了一致好评。



图1.8知乎网主页的推荐

## 1.2.6 其他推荐的应用

以上五个模块是最为我们熟知，应用最为广泛的推荐算法应用模块。然而今年来随时互联网的飞速发展和深入，推荐算法的模块远远不止这五块。在这里由于篇幅原因，简单介绍一下目前其他推荐算法的应用模块

1. 个性化邮件
2. 个性化广告
3. 基于位置的个性化推荐
4. 基于语音的个性化推荐

# 1.3 推荐系统的相关技术

和搜索引擎不同，个性化推荐系统需要依赖用户的行为数据，因此一般都是作为一个应用存在于不同网站之中。在互联网的各类网站中都可以看到推荐系统的应用，而个性化推荐系统在这 些网站中的主要作用是通过分析大量用户行为日志，给不同用户提供不同的个性化页面展示，来 提高网站的点击率和转化率。

目前主要的推荐方法有：

## 1.3.1 基于内容推荐

基于内容的推荐（Content-based Recommendation）是[4]  [信息过滤](https://baike.baidu.com/item/%E4%BF%A1%E6%81%AF%E8%BF%87%E6%BB%A4)技术的延续与发展，它是建立在项目的内容信息上作出推荐的，而不需要依据用户对项目的评价意见，更多地需要用机 器学习的方法从关于内容的特征描述的事例中得到用户的兴趣资料。在基于内容的推荐系统中，项目或对象是通过相关的特征的属性来定义，系统基于用户评价对象 的特征，学习用户的兴趣，考察用户资料与待预测项目的相匹配程度。用户的资料模型取决于所用学习方法，常用的有决策树、神经网络和基于向量的表示方法等。 基于内容的用户资料是需要有用户的历史数据，用户资料模型可能随着用户的偏好改变而发生变化。

基于内容推荐方法的优点是：  
　　1）不需要其它用户的数据，没有冷开始问题和稀疏问题。  
　　2）能为具有特殊兴趣爱好的用户进行推荐。  
　　3）能推荐新的或不是很流行的项目，没有新项目问题。  
　　4）通过列出推荐项目的内容特征，可以解释为什么推荐那些项目。  
　　5）已有比较好的技术，如关于分类学习方面的技术已相当成熟。

## 1.3.2协同过滤推荐

[协同过滤](https://baike.baidu.com/item/%E5%8D%8F%E5%90%8C%E8%BF%87%E6%BB%A4)推荐（Collaborative Filtering Recommendation）技术是推荐系统中应用最早和最为成功的技术之一。它一般采用最近邻技术，利用用户的历史喜好信息计算用户之间的距离，然后 利用目标用户的最近邻居用户对商品评价的加权评价值来预测目标用户对特定商品的喜好程度，系统从而根据这一喜好程度来对目标用户进行推荐。协同过滤最大优 点是对推荐对象没有特殊的要求，能处理非结构化的复杂对象，如音乐、电影。

基于[协同过滤](https://baike.baidu.com/item/%E5%8D%8F%E5%90%8C%E8%BF%87%E6%BB%A4)的推荐系统可以说是从用户的角度来进行相应推荐的，而且是自动的即用户获得的推荐是系统从购买模式或浏览行为等隐式获得的，不需要用户努力地找到适合自己兴趣的推荐信息，如填写一些调查表格等。

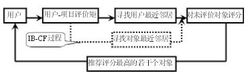


图1.9 协同过滤算法

图1.9 基于用户的系统过滤推荐过程

和基于内容的过滤方法相比，协同过滤具有如下的优点：  
　　1） 能够过滤难以进行机器自动内容分析的信息，如艺术品，音乐等。  
　　2） 共享其他人的经验，避免了内容分析的不完全和不精确

## 1.3.3 基于关联规则推荐

基于关联规则的推荐（Association Rule-based Recommendation）是以关联规则为基础，把已购商品作为规则头，规则体为推荐对象。关联规则挖掘可以发现不同商品在销售过程中的相关性，在零 售业中已经得到了成功的应用。管理规则就是在一个交易数据库中统计购买了商品集X的交易中有多大比例的交易同时购买了商品集Y，其直观的意义就是用户在购 买某些商品的时候有多大倾向去购买另外一些商品。比如购买牛奶的同时很多人会同时购买面包。

算法的第一步关联规则的发现最为关键且最耗时，是算法的瓶颈，但可以离线进行。其次，商品名称的同义性问题也是关联规则的一个难点。

## 1.3.4 基于效用推荐

基于效用的推荐（Utility-based Recommendation）是建立在对用户使用项目的效用情况上计算的，其核心问题是怎么样为每一个用户去创建一个效用函数，因此，用户资料模型很大 程度上是由系统所采用的效用函数决定的。基于效用推荐的好处是它能把非产品的属性，如提供商的可靠性（Vendor Reliability）和产品的可得性（Product Availability）等考虑到效用计算中。

## 1.3.5 基于知识推荐

基于知识的推荐（Knowledge-based Recommendation）在某种程度是可以看成是一种推理（Inference）技术，它不是建立在用户需要和偏好基础上推荐的。基于知识的方法因 它们所用的功能知识不同而有明显区别。效用知识（Functional Knowledge）是一种关于一个项目如何满足某一特定用户的知识，因此能解释需要和推荐的关系，所以用户资料可以是任何能支持推理的知识结构，它可以 是用户已经规范化的查询，也可以是一个更详细的用户需要的表示。

## 1.3.6 组合推荐

由于各种推荐方法都有优缺点，所以在实际中，组合推荐（Hybrid Recommendation）经常被采用。研究和应用最多的是内容推荐和协同过滤推荐的组合。最简单的做法就是分别用基于内容的方法和协同过滤推荐方法 去产生一个推荐预测结果，然后用某方法组合其结果。尽管从理论上有很多种推荐组合方法，但在某一具体问题中并不见得都有效，组合推荐一个最重要原则就是通 过组合后要能避免或弥补各自推荐技术的弱点。

在组合方式上，有研究人员提出了七种组合思路：  
　　1）加权（Weight）：加权多种推荐技术结果。  
　　2）变换（Switch）：根据问题背景和实际情况或要求决定变换采用不同的推荐技术。  
　　3）混合（Mixed）：同时采用多种推荐技术给出多种推荐结果为用户提供参考。  
　　4）特征组合（Feature combination）：组合来自不同推荐数据源的特征被另一种推荐算法所采用。  
　　5）层叠（Cascade）：先用一种推荐技术产生一种粗糙的推荐结果，第二种推荐技术在此推荐结果的基础上进一步作出更精确的推荐。  
　　6）特征扩充（Feature augmentation）：一种技术产生附加的特征信息嵌入到另一种推荐技术的特征输入中。  
　　7）元级别（Meta-level）：用一种推荐方法产生的模型作为另一种推荐方法的输入。

## 1.3.7 基于深度学习的推荐技术

其实，上面所讲协同过滤的方法是一种比较传统的方式，仍旧在工业界具有广泛的应用。如今，伴随着机器学习的兴起了非常多的技术被应用到推荐系统中，从传统的机器学习方式LR、GBDT、XGBoost到LightGBM，深度学习从最初利用word2vec用于评估用户的相似度，到CNN、RNN等模型也开始被很多的推荐小组尝试。

深度学习具有优秀的自动提取特征的能力，能够学习多层次的抽象特征表示，并对异质或跨域的内容信息进行学习，可以一定程度上处理推荐系统冷启动问题。

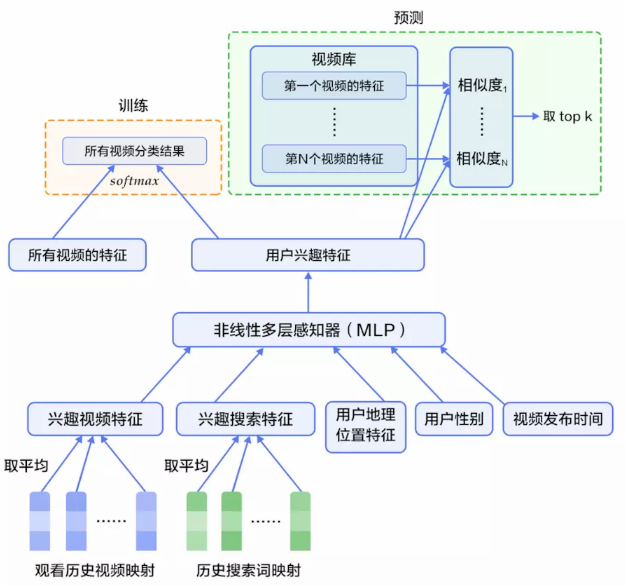


图1.10 Youtube视频的融合推荐模型

# 1.4 实践成果背景

目前几乎在各种各样的网站上都可以看到五花八门的推荐系统: 电子商务网站…视频网站…博客网站… 在学习中需要经常使用的论文网站却至今没有出现较为成熟的推荐系统, 学习中引用论文是一件普遍的事情，但就目前来看, 论文分类众多，数量庞大，查找并筛选同类论文成了一种繁琐的事情, 分类和搜索两种常用的解决办法很多时候不能很好的解决用户需求。

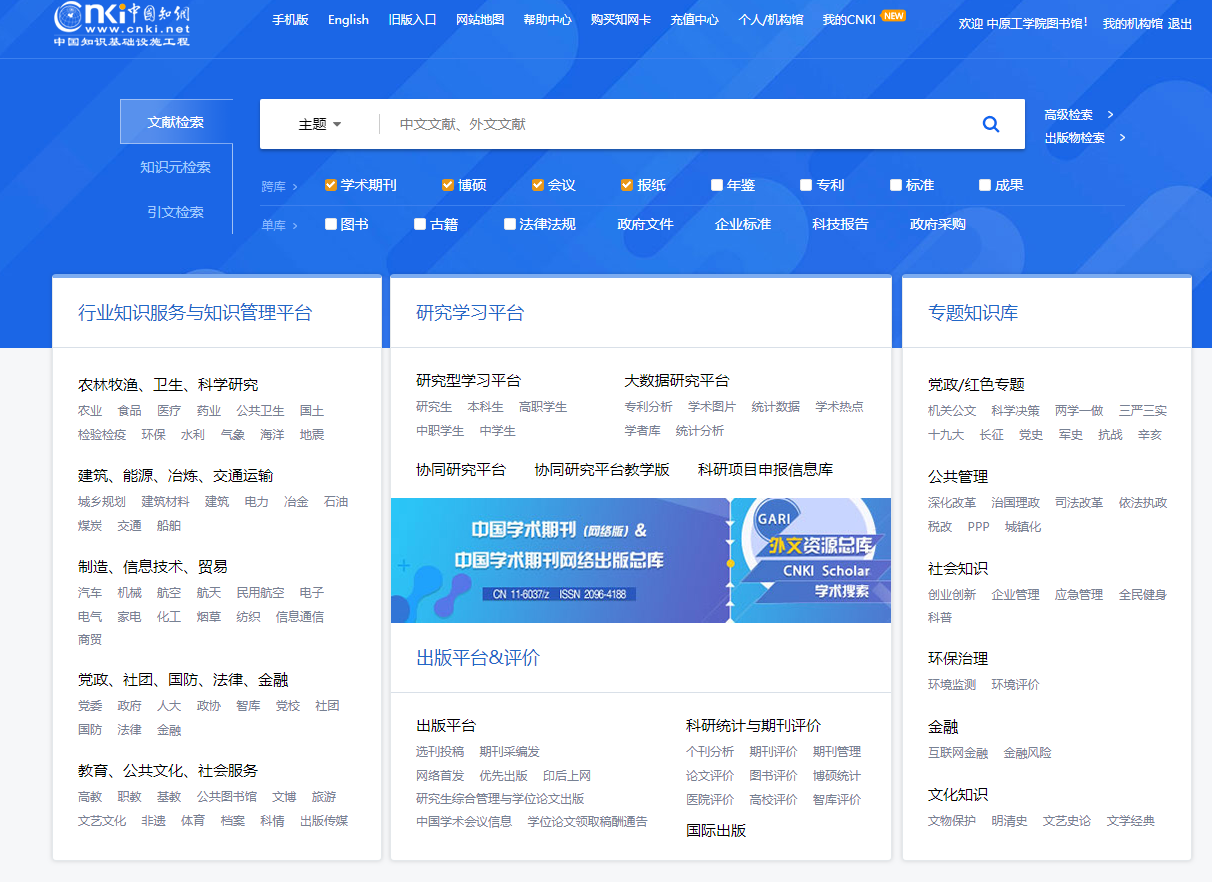


图1.11 中国知网

就论文网站的现状来看，每年产生的各类论文参差不齐，数量常常数以万记。而我们作为学生尤其是普通本科生在查找文献翻阅论文时时常难以去揣摩把我论文质量的好坏和是否为我们所需。如果仅凭分类和搜索查找，对于每一篇论文我们都需要阅读，仅仅算是大致浏览的话，也是一项时间成本极为浩大的工作，在这种情况下，我们需要一个推荐系统能够根据我们的需求，结合我们的行为和论文的特征去动态的为我们推荐一些论文，从而达到提升工作效力和生产力的目的。

在实践成果三中, 我和我的搭档邓琳在贾晓辉老师帮助下, 顺利完成了论文推荐系统的原型图以及业务逻辑和框架, 界面以及登陆, 注册, 搜素, 分类, 数据库等网站的基础功能并初步实现了**基于物品的协同过滤算法**( item-based collaborative filtering ).

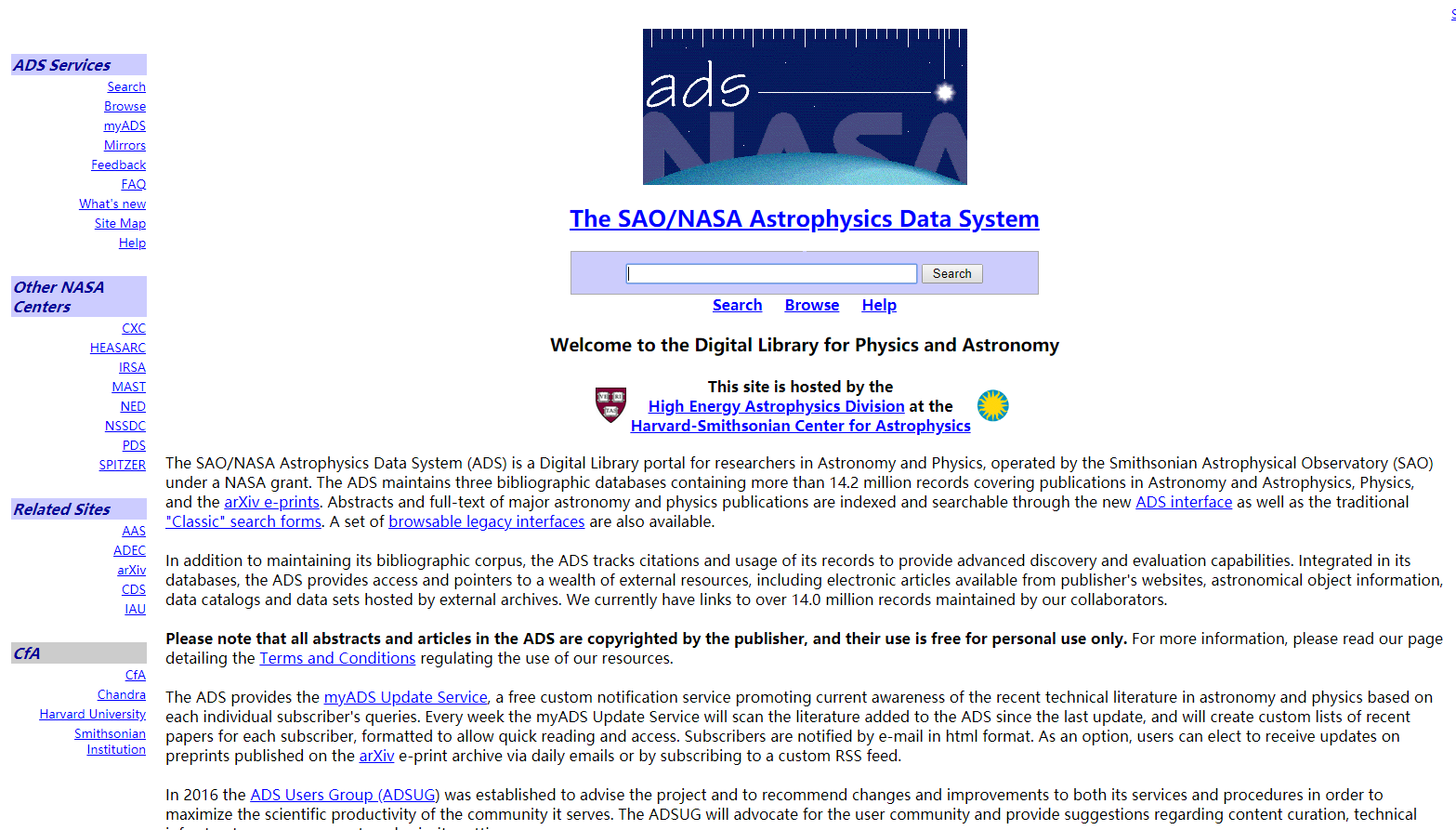


图 1.12 国外论文网站

第二章 需求分析

# 2.1 业务分析

论文推荐网站本质上依然是论文管理网站，网站的主要信息就是论文，主要的用户对象就是需要查阅论文的各大高校的学生及教职工。这一群体在论文查阅的需求上比较单一，只是学习和工作需求，不掺杂娱乐以及社交需求。

因此在外观设计上和功能上应该符合这一群体的基本要求。界面要做到简洁美观，拒绝繁杂绚丽。在功能上应该做到基本功能齐全，避免过多繁杂的功能。

在信息载体方面，因为近年来信息的爆炸式发展，论文的数量也在不断增加。在数据库设计上应该具有一定的前瞻性，对未来可预见到的数据危险进行提前的处理。

在算法选择上应该尽量选择较为高效便捷，又能够迅速反应的算法。

在推荐引擎上，同样应该采用高效迅速的引擎，同时避免数据的过于冗余。

# 2.2 系统功能需求分析

在功能上考虑基本的需求功能，同时参考了《中国知网》的布局和设计，主要应该有以下功能：

账号登录&注册功能：用户可以在网站填写信息并注册账号，注册后可以直接登录账号等；

公告栏，可以显示各种公告信息如网站公告等

排行栏：可以依据论文热度排行，最新论文等

轮播图：动态的显示大事件以及重要公告

5、分类功能：依据不同学科，热度等特征预先将论文进行分类，方便用户直接进行查找。

6、搜索功能：在搜索框中直接输入论文相关信息，如题目，作者，内容等可以直接在数据库中查询并返回搜索结果。

7、推荐功能:系统根据用户搜索的内容进行推荐，向用户推荐其可能感兴趣的论文；

8、评分功能：因为采用了协同过滤算法需要计算用户之间的相似度，为了便于实现推荐系统需要增加评分功能。

# 2.3 系统性能需求分析

考虑到目前普遍的电脑性能以及服务器性能和成本问题，在性能问题上应该保证用户体验不受损的前提下尽量减少数据冗余，避免数据库的复杂化。

1、系统处理的准确性和及时性

系统处理的准确性和及时性是系统的必要性能。在系统设计和开发过程中，要充分考虑系统当前和将来可能承受的工作量，能够存储足够多的论文，能够准确的让用户管理自己的信息、收藏、发表的论文以及对推荐的评论，并且能够快速准确的显示用户搜索的论文信息，能够快速、及时的推荐。总的来说就是系统的处理能力和响应时间能够满足用户对系统的需求。

2、系统的开放性和系统的课扩充性

论文推荐系统在开发过程中，应当充分考虑以后的可扩充性。而要实现这点,应通过系统的开放性来完成,既系统应是一个开放系统,只要符合一定的规范,可以简单的加入和减少系统的模块,配置系统的硬件。通过软件的修补、替换完成系统的升级和更新换代。

3、系统的易用性和易维护性

为了让论文推荐系统的使用用户简单方便的使用。这要求系统能够提供简单的使用方法,系统的功能比较基本,所以用户使用起来会比较方便;针对用户可能出现的使用问题,要提供足够的帮助,缩短用户对系统熟悉的过程。

# 2.4系统数据需求分析

论文推荐系统的数据需求包括以下几点：

1、数据录入和处理的准确性

数据的输入是否准确是数据处理的前提,错误的输入会导致系统输出的不正确和不可用,从而使系统的工作失去意义。数据的输入来源是手工输入。手工输入要通过系统界面上的安排系统具有容错性,用户根据提示进行正确的使用,而降低出错概率。

2、数据的一致性和完整性

用户搜索论文的信息与发表的论文信息要保持一致性和完整性，不能在不同的界面显示的信息不同。

3、数据的安全性

在处理推荐算法时必须要经常调用数据库中用户的数据来进行算法分析和设计，从而达到动态智能推荐的目的。但很重要的一点就是一定要保证数据在数据库中的安全性，才能避免出错。

第3章 系统设计

# 3.1 系统功能设计

“中工知网”是一款Web端系统，能够支持国内外大多的主流浏览器。系统采用传统的MVC开发模式，即模型（Model）、试图（View）、控制器(Controller)。

下面主要介绍”中工知网”Web端网站的线上功能，功能如下：

1、账号登录、注册功能：用户可以在网站填写信息并注册账号，注册后可以直接登录账号等；

2、主界面功能：主界面是诸多功能的连接点，应该符合人们的审美标准，简洁漂亮。同时主页面应该包含以下基本信息：用户，搜索功能面板，论文分类栏，公告栏，轮播图，相关资讯，排行榜以及智能推荐栏等；

3、分类功能：依据不同学科，热度等特征预先将论文进行分类，方便用户直接进行查找。

4、搜索功能：在搜索框中直接输入论文相关信息，如题目，作者，内容等可以直接在数据库中查询并返回搜索结果。

5、推荐功能:系统根据用户搜索的内容进行推荐，向用户推荐其可能感兴趣的论文；

6、反馈喜欢功能：因为采用了协同过滤算法需要计算用户之间的相似度，为了便于实现推荐系统需要增加评分功能。

网站的基本操作实现了在功能分析阶段罗列出的相关功能并完整的定义了各个功能之间的联系以及与数据库的交互。

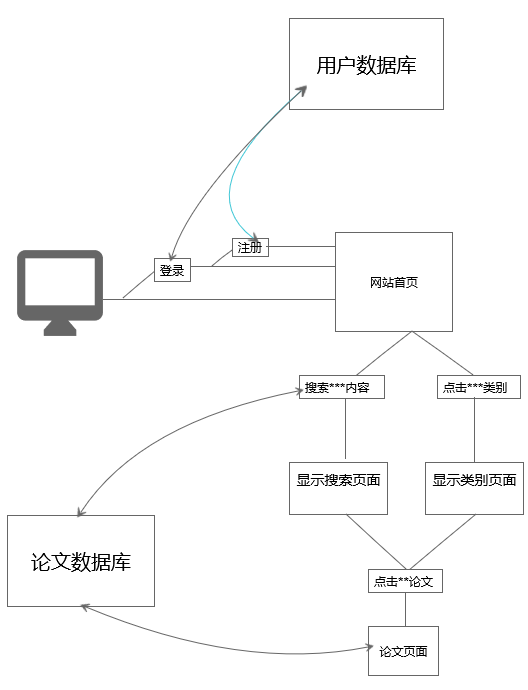


图3.1 系统流程设计

# 3.2算法设计

“中工知网”论文管理系统的核心在于推荐系统，这也是本系统相比于其他论文管理系统的亮点所在。“中工知网”论文管理系统采用了目前最主流的推荐算法——基于物品的协同过滤算法。

下面着重介绍一下本系统所采用的的算法基于物品的协同过滤算法和为何使用该算法

## 3.2.1 用户行为分析

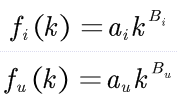
在利用用户行为数据设计推荐算法之前，研究人员首先需要对用户行为数据进行分析，了解数据中蕴含的一般规律，这样才能对算法的设计起到指导作用。本节将介绍用户行为数据中蕴含 的一般规律，这些规律并不是只存在于一两个网站中的特例，而是存在于很多网站中的普遍规律。本系统自然也不例外。

很多关于互联网数据的研究发现，互联网上的很多数据分布都满足一种称为Power Law的分布，这个分布在互联网领域也称**长尾分布**。



长尾分布其实很早就被统计学家注意到了。1932年，哈佛大学的语言学家Zipf在研究英文单词的词频时发现，如果将单词出现的频率按照由高到低排列，则每个单词出现的频率和它在热门 排行榜中排名的常数次幂成反比。这个分布称为Zipf定律。这个现象表明，在英文中大部分词的词频其实很低，只有很少的词被经常使用。

很多研究人员发现，用户行为数据也蕴含着这种规律。令为对k个物品产生过行为的用户数，令为被k个用户产生过行为的物品数。那么，和都满足长尾分布。也就是说：



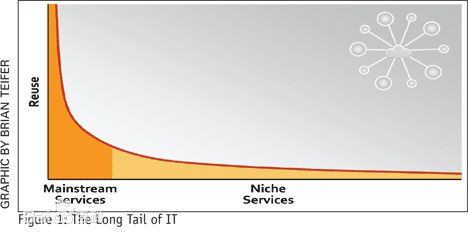
我个人对长尾分布的理解就是：常用的极其常用，少用的几乎不用。在本系统中，也一定程度的存在这种情况：少部分比较受欢迎、火爆的论文几乎被所有人都“喜欢”了。而大部分或者说一部分比较冷门，鲜有人浏览的论文又几乎没有人去“喜欢”，这样就导致了可能给所有人的推荐论文列表都存着类似的情况——因为大家的“喜欢情况”存在长尾分布。在后面我将提到我是如何在本系统中解决这个问题。

图3.2 长尾分布

## 3.2.2 协同过滤算法

仅仅基于用户行为数据设计的推荐算法一般称为协同过滤算法。学术界对协同过滤算法 进行了深入研究，提出了很多方法，比如基于邻域的方法（neighborhood-based）、隐语义模型（latent factor model）、基于图的随机游走算法（random walk on graph）等。在这些方法中， 最著名的、在业界得到最广泛应用的算法是基于邻域的方法，而基于邻域的方法主要包含下面两种算法。

 基于用户的协同过滤算法 这种算法给用户推荐和他兴趣相似的其他用户喜欢的物品。

 基于物品的协同过滤算法 这种算法给用户推荐和他之前喜欢的物品相似的物品。

基于用户行为分析的推荐算法是个性化推荐系统的重要算法，学术界一般将这种类型的算法称为协同过滤算法。顾名思义，协同过滤就是指用户可以齐心协力，通过不断地和网站互动，使 自己的推荐列表能够不断过滤掉自己不感兴趣的物品，从而越来越满足自己的需求

协同过滤算法是基于用户行为数据分析设计的，本系统中用户行为数据的表现形式为用户对某篇论文的”喜欢”，表明该用户对某篇论文的喜好。该数据储存在一张like表中，一个用户id对应一个喜欢论文的id。

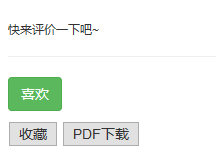


图3.3 系统反馈按钮

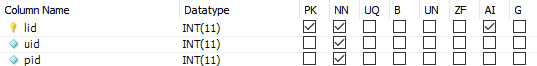


图3.4 数据表

用户行为在个性化推荐系统中一般分两种——显性反馈行为（explicit feedback）和隐性反馈行为（implicit feedback）。显性反馈行为包括用户明确表示对物品喜好的行为。本系统所采用的是显性反馈行为，即用户对一篇论文表示“喜欢”，明确的喜好行为。本系统没有将用户的浏览记录作为隐形反馈行为的原因就是考虑到，在显性反馈中，很容易区分一个用户行为是正反馈还是负反馈， 而在隐性反馈行为中，就相对比较难以确定。

协同过滤算法是基于邻域实现的，是推荐系统中最基础的算法，该算法不仅在学术界得到了深入研究，而且在业界得到了广泛应用。本系统的数据集较为基础，功能量和用户数都比较少，故采用了协同过滤算法作为推荐系统的算法。

## 3.2.3 两种协同过滤算法对比

基于邻域的算法是推荐系统中最基本的算法，该算法不仅在学术界得到了深入研究，而且在业界得到了广泛应用。基于邻域的算法分为两大类，一类是基于用户的协同过滤算法，另一类是 基于物品的协同过滤算法。下面将对这两种算法进行讨论，对比它们的优缺点。

## 3.2.4 基于用户的协同过滤算法

基于用户的协同过滤算法是推荐系统中最古老的算法。可以不夸张地说，这个算法的诞生标志了推荐系统的诞生。该算法在1992年被提出，并应用于邮件过滤系统，1994年被GroupLens用 于新闻过滤。在此之后直到2000年，该算法都是推荐系统领域最著名的算法。本节将对该算法进 行详细介绍，首先介绍最基础的算法，然后在此基础上提出不同的改进方法，并通过真实的数据 集进行评测

基于用户的协同过滤算法主要包括两个步骤。

(1) 找到和目标用户兴趣相似的用户集合。

(2) 找到这个集合中的用户喜欢的，且目标用户没有听说过的物品推荐给目标用户。

步骤(1)的关键就是计算两个用户的兴趣相似度。这里，协同过滤算法主要利用行为的相似度 计算兴趣的相似度。给定用户u和用户v，令N(u)表示用户u曾经有过正反馈的物品集合，令N(v) 为用户v曾经有过正反馈的物品集合。那么，我们可以通过如下的Jaccard公式简单地计算u和v的 兴趣相似度：



或者通过余弦相似度计算



该代码对两两用户都利用余弦相似度计算相似度。这种方法的时间复杂度是O(|U|\*|U|)，这在用户数很大时非常耗时。事实上，很多用户相互之间并没有对同样的物品产生过行为，即很多时候| N(u)∩N(v) | = 0 。上面的算法将很多时间浪费在了计算这种用户之间的相似度上。如果换一 个思路，我们可以首先计算出| N(u)∩N(v) | ≠ 0的用户对，然后再对这种情况除以分母。



针对我们的系统来进行考虑的话，由于我们的数据集较为单一，用户量和论文量也很小，如果采用基于用户的协同过滤算法可能会导致冷启动，重复推荐等一系列问题，所以很难完全根据用户来进行推荐，因此本系统没有采用此推荐算法。

## 3.2.5 基于物品的协同过滤算法

基于物品的协同过滤（item-based collaborative filtering）算法是目前业界应用最多的算法。无论是亚马逊网，还是Netflix、Hulu、YouTube，其推荐算法的基础都是该算法。本节将从基础 的算法开始介绍，然后提出算法的改进方法，并通过实际数据集评测该算法

基于用户的协同过滤算法在一些网站（如Digg）中得到了应用，但该算法有一些缺点。首先，随着网站的用户数目越来越大，计算用户兴趣相似度矩阵将越来越困难，其运算时间复杂度和空 间复杂度的增长和用户数的增长近似于平方关系。其次，基于用户的协同过滤很难对推荐结果作 出解释。因此，著名的电子商务公司亚马逊提出了另一个算法——基于物品的协同过滤算法① 基于物品的协同过滤算法（简称ItemCF）给用户推荐那些和他们之前喜欢的物品相似的物品。。

比如，该算法会因为你购买过《数据挖掘导论》而给你推荐《机器学习》ItemCF算法并不利用物品的内容属性计算物品之间的相似度，它主要通过分析用户的行为记录计算物品之间的 相似度。该算法认为，**物品A和物品B具有很大的相似度是因为喜欢物品A的用户大都也喜欢物品 B**。

基于物品的协同过滤算法主要分为两步。

(1) 计算物品之间的相似度。

(2) 根据物品的相似度和用户的历史行为给用户生成推荐列表。

下面的公式定 义物品的相似度：



这里，分母|N(i)|是喜欢物品i的用户数，而分子 () ( ) Ni N j 是同时喜欢物品i和物品j的用户数。因此，上述公式可以理解为喜欢物品i的用户中有多少比例的用户也喜欢物品j。

上述公式虽然看起来很有道理，但是却存在一个问题。如果物品j很热门，很多人都喜欢，那么Wij就会很大，接近1，这也就是前面我们提到的长尾分布。因此，该公式会造成任何物品都会和热门的物品有很大的相似度，这 对于致力于挖掘长尾信息的推荐系统来说显然不是一个好的特性。为了避免推荐出热门的物品，我们需要对该公式进行改进：



这个公式惩罚了物品j的权重，因此减轻了热门物品会和很多物品相似的可能性。 从上面的定义可以看到，在协同过滤中两个物品产生相似度是因为它们共同被很多用户喜欢，也就是说每个用户都可以通过他们的历史兴趣列表给物品“贡献”相似度。这里面蕴涵着一 个假设，就是每个用户的兴趣都局限在某几个方面，因此如果两个物品属于一个用户的兴趣列表， 那么这两个物品可能就属于有限的几个领域，而如果两个物品属于很多用户的兴趣列表，那么它 们就可能属于同一个领域，因而有很大的相似度。 和UserCF算法类似，用ItemCF算法计算物品相似度时也可以首先建立用户—物品倒排表（即

对每个用户建立一个包含他喜欢的物品的列表），然后对于每个用户，将他物品列表中的物品两 两在共现矩阵C中加1。

在本系统中就是基于上面的公式实现的，简单来说，就是对于当前用户的喜欢列表中 **每一个** 喜欢的物品, 所有用户中 **喜欢了这个物品** 又喜欢了其他的物品中 相似度最高的几个物品。实现方法就是当前用户的喜欢列表 与 用户 i 的喜欢列表 构建 当前矩阵所有 当前矩阵 累加得到共现矩阵。如果出现了推荐数量不足或用户未登录的情况，我们就采用最基础的TopN推荐，即用最火爆的几个推荐补足推荐数，从而也解决了冷启动的问题。

代码的实现我将在下一章提到，下面是基于物品的协同过滤算法在本系统中的程序逻辑框图。



图3.5 程序框图

# 3.4 数据库设计

为了方便用户的操作以及推荐的调用，系统采用了模块化设计的思想，通过各模块的不同组合以实现不同的增、删、改、查功能。最终，使该系统能够具有规范化的格式、完整的信息量和易于查询等的特点。

## 3.4.1 数据库的详细设计

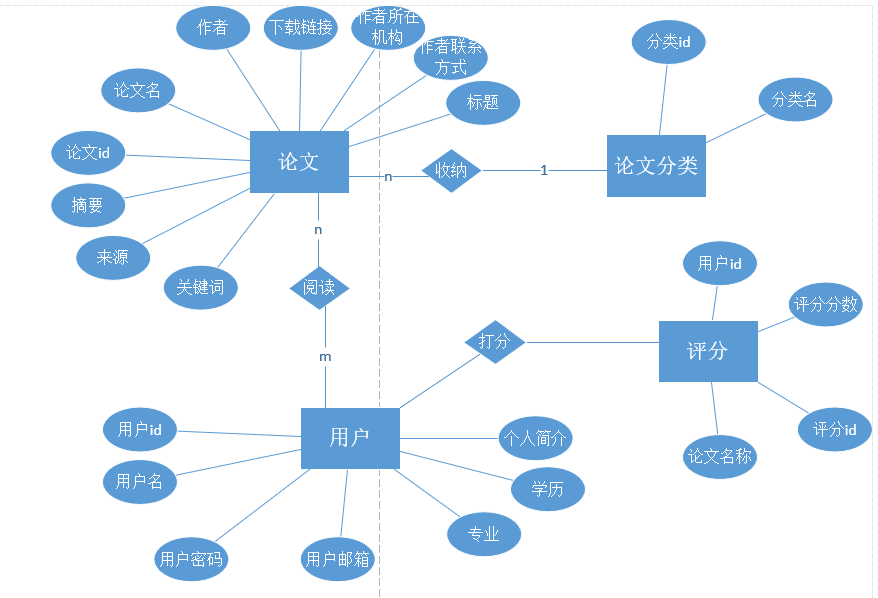
依据数据库的概要设计，最后总结出数据库结构的主要数据表包括:读者表，图书表，管理员表，读者用户表，管理员用户表等。以下将给出系统数据库设计的逻辑模型，即各数据表的结构

图 3.6 数据库设计1

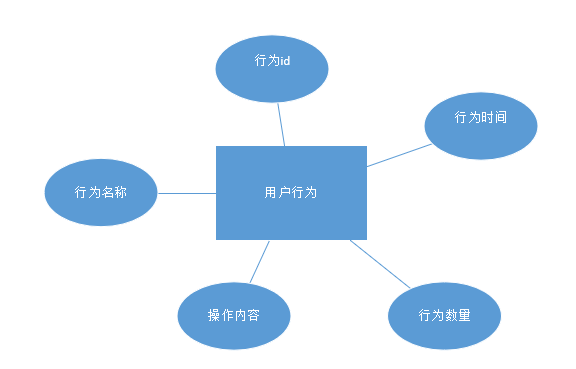


图 3.7 数据库设计2

1、用户表

用户表存储系统客户的相关信息，表结构如表4.1所示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 说明 | 数据类型 | 长度 | 空 | 主键 |
| uid | 用户号 | nvarchar | 0 | 否 | 是 |
| username | 用户名 | nvarchar | 0 | 是 | 否 |
| email | 用户邮箱 | nvarchar | 20 | 是 | 否 |
| major | 专业 | nvarchar | 10 | 是 | 否 |
| education | 学历 | nvarchar | 20 | 是 | 否 |
| introduce | 个人简介 | nvarchar | 50 | 是 | 否 |
| interest | 兴趣爱好 | nvarchar | 20 | 是 | 否 |
| password | 密码 | nvarchar | 10 | 是 | 否 |

表3.1 用户表结构

2、论文表

论文表存储论文的相关信息，表结构如表4.2所示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 说明 | 数据类型 | 长度 | 允许空 | 是否为主键 |
| aid | 论文id | nvarchar | 10 | 否 | 是 |
| anmae | 论文名 | nvarchar | 10 | 是 | 否 |
| author | 作者 | nvarchar | 10 | 是 | 否 |
| insititution | 作者所在机构 | nvarchar | 10 | 是 | 否 |
| aemail | 作者联系方式 | nvarchar | 20 | 是 | 否 |
| source | 来源 | nvarchar | 20 | 是 | 否 |
| title | 标题 | nvarchar | 20 | 是 | 否 |
| downurl | 下载链接 | nvarchar | 50 | 是 | 否 |
| abstract | 摘要 | nvarchar | 50 | 是 | 否 |
| keyword | 关键词 | nvarchar | 20 | 是 | 否 |

表3.2 论文表结构

3、论文分类表

论文分类表存储论文类别，表结构如表4.3所示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 说明 | 数据类型 | 长度 | 允许空 | 是否为主键 |
| cid | 类别id | int | 10 | 否 | 是 |
| cname | 类别名字 | nvarchar | 10 | 否 | 否 |

表3.3 论文分类表结构

4、用户行为表

用户行为表(推荐相关)的相关信息，表结构如表4.4所示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 说明 | 数据类型 | 长度 | 允许空 | 是否为主键 |
| uaid | 行为id | int | 10 | 否 | 是 |
| action | 行为名(共有down, click, search三种) | nvarchar | 10 | 是 | 否 |
| content | 行为内容(共有下载内容, 搜索内容, 点击内容三种) | nvarchar | 30 | 是 | 否 |
| count | 行为次数 | int | - | 是 | 否 |
| uatime | 行为时间 | DateTime | - | 是 | 否 |

表3.4用户行为表结构

3、论文评分表（推荐相关）

和推荐相关的论文评分表，表结构如表4.3所示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 说明 | 数据类型 | 长度 | 允许空 | 是否为主键 |
| rsid | 评分id | int | 10 | 否 | 是 |
| ruid | 用户id | int | 10 | 否 | 否 |
| rpid | 论文id | int | 10 | 否 | 否 |

表3.5 论文分类表结构

# 第4章 系统实现

本章我们将实现一章提出的功能，并对后端的关键模块给出核心代码

# 4.1 系统界面实现

本章主要说明了系统的实现过程，以及主要的功能模块，是论文推荐系统的核心章节，本章节的内容主要是从系统界面的实现、功能模块的设计实现和核心算法三个方面，说明论文推荐系统的实现过程。

## 4.1.1 登录/注册界面

论文推荐系统的界面使用HTML5+CSS+JSP和Bootstrap框架实现，发布在服务器上，用户根据网站地址可以访问我们的系统。



图 4.1 登录界面



图 4.2 注册界面

## 4.1.2 主页面

主页面是网站的默认页面，也是各个页面的导航页面。主要有以下几块。

主界面显示导航分类栏，点击即刻跳转到对应分类；搜索框，用户可以进行论文搜索，把自己想要看的论文搜索出来，搜索的格式为搜索类型+搜索内容；轮播图，显示当前活动公告以及推荐栏；知网动态显示最新文章等；火爆推荐显示最近最火热的一些论文；还有两个和用户行为相关的收藏的内容和浏览的内容，便于用户查找；最下面的是推荐栏可以显示推荐算法输出的推荐内容。



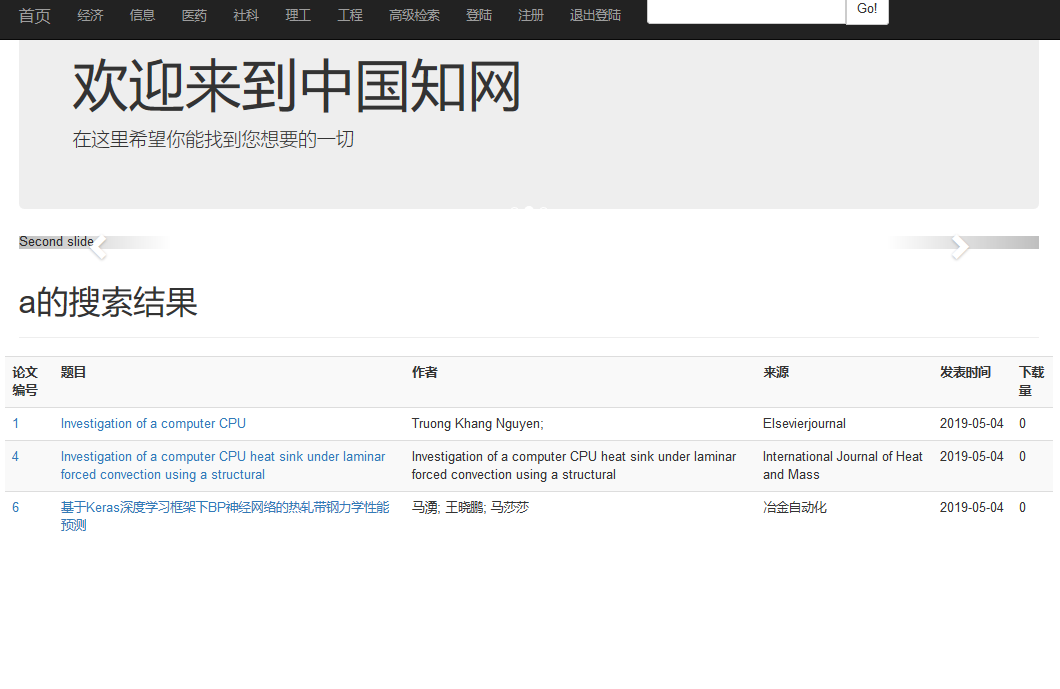
图 4.3 主界面

## 4.1.3 搜索/分类显示页面

在搜索栏中输入内容并点击搜索，或直接点击上面的导航分类页面后就会跳转到各个分类或搜索的显示页面。这个页面显示用户在主页面搜索框内搜索的内容。其中显示论文的ID，标题，作者，发表单位，文献来源，关键字，发表时间，第一负责人，点击量等。



图4.4 分类页面

图 4.5 搜索页面

## 4.1.4 论文阅读页面

在任何地方显示的论文超链接处点击蓝字的论文标题就会跳转到每个论文的显示页面。

此页面显示论文的具体信息。还包括下载，收藏，你可能想要的文章，对推荐的评价等功能。这些功能可以让读者实现一系列操作，满足一些用户需求。这个页面就是想为读者展现论文的具体信息，让读者可以根据这些信息对论文进行操作，如喜欢这篇论文可以进行收藏，若是对我们的推荐有想法、意见，也可以在下方进行评价。这个页面，我们是参考知网进行设计的，知网在论文网站这一领域走在前沿，我们希望从中找到其成功的原因、方法。



图 4.6 论文界面

# 功能模块实现

下面为本系统的核心模块的代码展示，论文推荐系统的核心功能模块有注册\登陆、智能推荐、主页、搜索这三个模块组成。

## 4.2.1 注册\登陆模块

但用户访问网站时先判断用户是否登陆，如果没有登录则需要先进行注册

主要使用了JavaSE、JSP、MySQL和Servlet技术来实现，相关核心代码如下。

package servlet;  
  
import …;  
  
@WebServlet(name = "LoginServlet", urlPatterns = "/LoginServlet")  
public class LoginServlet extends javax.servlet.http.HttpServlet {  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 response.setContentType("text/html;charset=utf-8");  
 String username = request.getParameter("username");  
 String password = request.getParameter("password");  
 UserDao dao = new UserDao();  
  
 //查看数据库中是否有该用户  
 User qUser = dao.findByName(username);  
 if(qUser==null){  
 request.setAttribute("status", "该用户不存在");  
 System.*out*.println("该用户不存在");  
 }else if(qUser.getPassword().equals(password)){  
 request.setAttribute("status", "登录成功, 等待跳转...");  
 System.*out*.println("登录成功, 等待跳转...");  
 request.getSession().setAttribute("user",qUser);  
 }else {  
 System.*out*.println("账号或密码输入错误");  
 request.setAttribute("status", "账号或密码输入错误");  
 }  
 response.sendRedirect("/zutki\_war\_exploded/IndexServlet");  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 doPost(request, response);  
 }  
}

package servlet;  
  
import …;  
  
@WebServlet("/RegistServlet")  
public class RegistServlet extends HttpServlet {  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 response.setContentType("text/html;charset=utf-8");  
 UserDao dao = new UserDao();  
  
 String username = request.getParameter("username");  
 String password = request.getParameter("password");  
 String email = request.getParameter("email");  
 String major = request.getParameter("major");  
  
 if(dao.isExist(username)){  
 request.setAttribute("message", "该用户已存在!");  
 }else{  
 User user = new User(username, password, major, email);  
 dao.userAdd(user);  
 request.setAttribute("message", "注册成功!");  
 }  
  
 response.setHeader("refresh", "3;url=/zutki\_war\_exploded/index.html"); //3秒后跳转到主页  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 doPost(request, response);  
 }  
}

package servlet;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet(name = "LogoutServlet", urlPatterns = "/LogoutServlet")  
public class LogoutServlet extends HttpServlet {  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 doGet(request, response);  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 request.getSession().removeAttribute("user");  
  
 response.sendRedirect("/zutki\_war\_exploded/IndexServlet");  
 }  
}

## 4.2.2搜索功能

搜索模块主要是用户在主页面的搜索框内输入搜索内容，系统根据内容在数据库进行查找，得到用户需要的数据，在将数据显示在搜索结果页面为用户进行展示。

package servlet;  
  
import dao.PaperDao;  
import entity.Paper;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.util.List;  
  
@WebServlet(name = "SearchServlet", urlPatterns = "/SearchServlet")  
public class SearchServlet extends HttpServlet {  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 doGet(request, response);  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 response.setContentType("text/html;charset=utf-8");  
 PaperDao dao = new PaperDao();  
  
 String classify = request.getParameter("classify");  
 String search = request.getParameter("search");  
  
 List<Paper> papers = dao.SearchPapers(classify, search);  
  
 System.*out*.println(papers.get(0).toString());  
 if(papers!=null){  
 Paper paper = papers.get(0);  
 request.setAttribute("title", paper.getTitle());  
 request.setAttribute("author", paper.getAuthor());  
 request.setAttribute("digest", paper.getDigest());  
 request.setAttribute("cnt", paper.getCnt());  
 request.setAttribute("date", paper.getDate());  
 request.setAttribute("downCnt", paper.getDownCnt());  
 request.setAttribute("place", paper.getPlace());  
 request.setAttribute("source", paper.getSource());  
 }else {  
 request.setAttribute("message", "找不到您输入的结果! 请重新查找");  
 }  
  
 request.setAttribute("papers", papers);  
 request.getRequestDispatcher("htmls/classify.jsp").forward(request, response);  
 return;  
 }  
}

## 4.2.3推荐模块

推荐模块主要实现当用户点击论文链接进入论文阅读界面时，通过用户点击论文的信息进行推荐，如根据用户点击论文的作者进行推荐、根据用户点击论文的关键字进行推荐以及根据用户点击论文的发表机构进行推荐等，核心代码如下。

package util;  
  
import dao.LikeDao;  
import dao.PaperDao;  
import dao.UserDao;  
import entity.Like;  
import entity.Paper;  
import entity.User;  
import java.util.\*;  
  
public class ItemSimilarity {  
 public static void main(String[] args) {  
 //测试小Demo  
 System.*out*.println("请输入用户希望推荐的用户id : ");  
 Scanner input = new Scanner(System.*in*);  
 int uid = input.nextInt();  
 ArrayList<Paper> v = *recommend*(uid);  
  
 System.*out*.println("正在生成针对用户id为"+uid+"的推荐...");  
 for(int i = 0; i < v.size(); i++)  
 System.*out*.println("第"+(i+1)+"个推荐: 题目:"+v.get(i).getTitle()+"");  
 }  
  
 //通过计算余弦相似度并取TopN, 返回为uid的用户生成的5个推荐文章  
 public static ArrayList<Paper> recommend(int uid){  
 UserDao userdao = new UserDao();  
 PaperDao paperdao = new PaperDao();  
 LikeDao likedao = new LikeDao();  
  
 ArrayList<Like> likeLists; //其他用户喜欢的论文列表  
  
 ArrayList<User> users = userdao.getAllUsers(); //所有用户列表  
 ArrayList<Paper> papers = paperdao.getAllPapers(); //所有论文列表  
 int[][] curMatrix = new int[papers.size()+5][papers.size()+5]; //当前矩阵  
 int[][] comMatrix = new int[papers.size()+5][papers.size()+5]; //共现矩阵  
 int[] N = new int[papers.size()+5]; //喜欢每个物品的人数  
  
 for(User user: users){  
 if(user.getUid()==uid) continue; //当前用户则跳过  
  
 likeLists = likedao.findLikesByUser(user.getUid()); //当前用户的喜欢列表  
  
 for(int i = 0; i < papers.size(); i++)  
 for(int j = 0; j < papers.size(); j++)  
 curMatrix[i][j] = 0; //清空矩阵  
  
 for(int i = 0; i < likeLists.size(); i++){  
 int pid1 = likeLists.get(i).getPid();  
 ++N[pid1];  
 for(int j = i+1; j < likeLists.size(); j++){  
 int pid2 = likeLists.get(j).getPid();  
 ++curMatrix[pid1][pid2];  
 ++curMatrix[pid2][pid1]; //两两加一  
 }  
 }  
 //累加所有矩阵, 得到共现矩阵  
 for(int i = 0; i < papers.size(); i++){  
 for(int j = 0; j < papers.size(); j++){  
 int pid1 = papers.get(i).getPid(), pid2 = papers.get(j).getPid();  
 comMatrix[pid1][pid2] += curMatrix[pid1][pid2];  
 comMatrix[pid1][pid2] += curMatrix[pid1][pid2];  
 }  
 }  
 }  
  
  
 TreeSet<Paper> preList = new TreeSet<Paper>(new Comparator<Paper>() {  
 @Override  
 public int compare(Paper o1, Paper o2) {  
 if(o1.getW()!=o2.getW())  
 return (int) (o1.getW()-o2.getW())\*100;  
 else  
 return o1.getCnt()-o2.getCnt();  
 }  
 }); //预处理的列表  
  
 likeLists = likedao.findLikesByUser(uid); //当前用户喜欢的论文列表  
 boolean[] used = new boolean[papers.size()+5]; //判重数组  
 for(Like like: likeLists){  
 int Nij = 0; //既喜欢i又喜欢j的人数  
 double Wij; //相似度  
 Paper tmp; //当前的论文  
  
 int i = like.getPid();  
 for(Paper paper: papers){  
 if(like.getPid() == paper.getPid()) continue;  
 int j = paper.getPid();  
  
 Nij = comMatrix[i][j];  
 Wij = (double)Nij/Math.*sqrt*(N[i]\*N[j]); //计算余弦相似度  
  
 tmp = paperdao.findPaperById(paper.getPid());  
 tmp.setW(Wij);  
  
 if(used[tmp.getPid()]) continue;  
 preList.add(tmp);  
 used[tmp.getPid()] = true;  
 }  
 }  
  
 ArrayList<Paper> recomLists = new ArrayList<>(); //生成的推荐结果  
 for(int i = 0; preList.size()>0 && i<5; i++){  
 recomLists.add(preList.pollLast());  
 preList.pollLast();  
 }  
 if(recomLists.size()<5){  
 //推荐数量不满5个, 补足喜欢数最高的文章  
 recomLists = paperdao.findTopNPapers(recomLists);  
 }  
  
 return recomLists;  
 }  
}

## 4.2.3 主页模块实现

package servlet;  
  
import dao.PaperDao;  
import entity.Paper;  
import entity.User;  
import util.ItemSimilarity;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.\*;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.Vector;  
  
@WebServlet(name = "IndexServlet", urlPatterns = "/IndexServlet")  
public class IndexServlet extends HttpServlet {  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 doGet(request, response);  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 response.setContentType("text/html;charset=utf-8");  
  
 PaperDao dao = new PaperDao();  
 HttpSession session = request.getSession();  
 User user = (User) session.getAttribute("user");  
  
 //生成推荐  
 ArrayList<Paper> recomends = new ArrayList<>();  
  
 if(user!=null)  
 recomends = ItemSimilarity.*recommend*(user.getUid());  
 else  
 recomends = dao.findTopNPapers(recomends);  
  
 request.setAttribute("recomends", recomends);  
  
 System.*out*.println("推荐列表为: ");  
 for(Paper paper: recomends)  
 System.*out*.println("推荐题目为: "+paper.getTitle());  
  
 request.getRequestDispatcher("htmls/index.jsp").forward(request, response);  
 }  
}

# 第5章 总结

软件开发是一项非常注重实践的工作。在平常的学习中，虽然我们陆陆续续完成了学校的很多相关课程，以及自己在私下也学习了很多的编程知识。但如果真正进行一个较为系统的软件开发，相当于要把自己之前零零散散的知识都穿成一个串，融会贯通运用起来，才能真正的意识到这是一门较为复杂的工作，需要大量的去查阅文档，去搜索学习。可以说一个软件开发的过程其实就是一个不断学习的过程。往往, 真正的困难只在不断地实践之中凸显出来, 而真正的工程开发能力也只能再亲身的开发中去实现去完善.

本系统的核心部分也就在算法，Web的搭建更多是基于之前二级课题的基础之上，这次对基于物品的协同过滤算法的实现让我设身处地理解了算法之美已经算法的伟大之处。本人是一名ACMer选手，这一年多以来的竞赛生涯帮助我领略了各种各样的经典基础算法，是算法竞赛的基础为我学习推荐算法打下了坚实的基础！

在算法之外，本次实验成果运用并实现了JavaWeb的相关技术如JavaEE，JSP，Servlet，MySQL，Bootstrap等技术以及核心的基于物品的协同过滤算法，通过本次的实践成果，我更加深刻的了解到了MVC模式，DAO模式，运用分层的技术可以提高开发效率。在开发界面显示部分时，仅仅需要考虑的是如何布局一个好的用户界面；开发模型时，仅仅要考虑的是业务逻辑和数据维护，这样能使开发者专注于某一方面的开发，提高开发效率。同时在该系统开发过程中，还学习了编码规范，让自己养成良好的代码规范。

在三级课题中，我初步实现了网站的原型和数据库框架，在后来的一学期中，我和我的搭档邓琳在贾晓辉老师的帮助下逐步实现并完善了整个网站的基本功能和简单的推荐系统算法。这两次为期一学期的开发工程让我在事件中明白了团队的作用，以及个人应该具备的团队沟通和协调能力。在一开始的开发过程中，由于我们对系统的认知不同，做出的东西往往差异很大，对系统的进展有所阻碍，不善于交流、沟通，也不知道到分工。后来我们两个在老师的帮助下改进了沟通办法，有意见就即时进行沟通，有分歧就立即进行分析，分工越来越明确，系统开发的进度也快了起来。

总之，这次的实践成果课题，我学到许多原来我并不知道或者仅仅知道不熟悉的知识，更通过实践深刻明白了开发系统过程中具体的工作流程，同时也了解到团队开发的重要性，在今后的学习过程中，我一定会再接再厉！

# 参考文献：

[1]Morgan928.实现推荐引擎系统

[J/OL].https://blog.csdn.net/u012216773/article/details/82667067#commentBox,2018年09月12日.

[2]xiazdong.JavaBean中DAO设计模式介绍

[J/OL].https://blog.csdn.net/xiazdong/article/details/6895465,2011年10月21日.

[3]传智播客，JavaWeb程序开发入门 [M]，北京：清华大学出版社， 2012.5

[4]传智播客，JavaWeb程序开发进阶[M]，北京：清华大学出版社， 2012.3

[5]何冰洁， 数据库原理与应用教程（第四版）[M],机械工业出版社

[6]项亮 推荐系统实践[M]，人民邮电出版社

[7] Resnick P,Iacovou N,Suchak M,etal.GroupLens:an open architecture for collaborative filtering of netnews[C]//Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work,Oct 22-26,1994.New York, NY, USA:ACM , 994:175-186.

指导教师评语

成绩：

指导教师签名：

年 月 日