基于深度学习的电影推荐系统的研究与实现

数学与信息学院　　软件工程专业
116052017051　　王浩轩　　指导教师：肖如良

【摘　要】个性化推荐不仅能提升用户的系统使用体验，还可为服务提供商带来巨大经济效益，具有重大研究价值。在大数据时代的今天，传统的协同推荐算法由于存在冷启动与排名损失问题，不能带来很好的用户体验。本文对前沿的深度推荐模型进行了充分的研究与实验，设计并实现了一个具有个性化推荐和热门推荐的混合电影推荐系统。在个性化推荐模块，本文选用了实验性能最优的 ENMF 模型。在热门推荐模块，本文将基于 sketch 的大规模流式数据频繁项统计技术和深度推荐模型进行混合，提出了基于个性化精排序的实时热门电影推荐方法，并选用了实验效果最优的 ItemPop-NCF 模型。本文通过爬虫对 MovieLens-1m 数据集进行了补全，经过测试，该混合框架系统的推荐功能准确、个性且多样，信息处理实时、稳定且高效。

【关键词】个性化推荐；热门推荐；混合推荐框架；深度学习

目录

1 绪论

1.1 研究背景

随着信息技术和互联网的飞速发展，数据量不断膨胀，所带来的是信息的负载[1]。对于信息的使用者，要想从海量的信息中快速地获取感兴趣的信息，是一件非常困难的事情。对于信息的提供方，如何最大化信息的投放价值，是他们所关心的，同时也是一件非常困难的事情。

在早期阶段，虽然分类目录（如 Hao123、2345、YAHOO 等）和搜索引擎（如 Bing、Google、百度等）的出现，从一定程度上缓解了信息过载所带来的上述困境，但这要求信息的使用者有明确的信息需求。当用户没有明确需求或无法准确的提炼检索关键词时，分类目录和搜索引擎也无能为力，可以说在一定程度上瘫痪了。为此，科学家们提出了推荐系统。与分类目录和搜索引擎一样，它可以快速帮助用户找到有用的信息，但不要求用户有明确的信息需求。另外，它可以基于用户主动建模，推荐用户可能感兴趣的信息，这与前者是互补的。目前，推荐系统已经成为众多电子商务网站的主要盈利工具，例如淘宝、亚马逊等，据统计亚马逊约有 20%~30%的收入来自于推荐系统[2]。许多音乐、视频应用软件也以精准推荐功能来增加其与同类应用软件竞争力，例如网易云音乐、QQ 音乐等的歌目推荐，优酷、爱奇艺、抖音、豆瓣等的影视推荐。

电影作为现代科技和艺术综合体。观看电影已经成为现代人不可缺少的一种娱乐方式。在紧凑的工作生活之余，看一部合胃口的优质电影，不失为一种极致的放松与娱乐。在如今信息过载的时代，电影资源同样也是，增长迅速。用户面对海量电影资源，无目的的选择，可能造成大量时间的浪费，从而失去耐心，产生厌烦，甚至放弃观影。如何让用户快速地找到想看的电影，减少挑选电影的耗时，个性化电影推荐至关重要。

从上世纪 90 年代以来，推荐技术层出不穷，有基于内容的推荐、协同过滤推荐、基于规则的推荐、基于效用的推荐、基于知识的推荐等，其中协同过滤推荐是应用最早且最成功的技术之一[1]。当今，深度学习发展迅速，其已在图像识别、自然语言处理等领域有着出色表现，但在推荐系统领域的应用仍处于初步阶段。为此，基于深度学习的推荐算法，深受学术界与工业界的关注，是当前的研究热点之一[2]。

1.2 国内外研究概况

1.2.1 国外研究现状

1992 年，Goldberg 等人在 Tapestry 邮件处理系统中首次引入了协同过滤的思想。在此后的 3 年内，相继出现的 Grouplens 新闻推荐系统、Ringo 音乐推荐系统及 Grundy 书籍推荐系统等，促进了协同推荐的发展[1]。

在 2006 年，Netflix 公司以 100 万美元的奖金，举办了著名的 Netflix Prize 推荐系统大赛[3]，吸引了众多优秀科学家的参与。在长达 3 年的比赛期间，参数者提出了相当多数量的推荐算法，极大地推动了推荐系统的发展。

2017 年，Alexandros 等人将深度学习首次引入基于内容的推荐系统中，引发学术界的关注[4]。如今，基于深度学习的推荐已经成为研究的热点。

1.2.2 国内研究现状

关于推荐系统的研究，国内起步较晚。2007 年，首个推荐系统研究团队“百分点”成立，并与工业界展开合作，提供实时智能推荐引擎服务。其后，各大互联网公司，如阿里巴巴、百度等，都慢慢成立了其各自的推荐团队，并不断发展壮大。字节跳动也因其抖音、今日头条产品的出色个性化推荐而备受关注。

2017 年，何向南等人将深度学习首次引入基于协同过滤的推荐系统中，提出广义矩阵分解模型及神经协同过滤模型[5]，为国内基于深度学习的协同过滤推荐奠定了基础。

近 3 年来，清华大学信息检索课题组（THUIR）首次将非采样策略应用到基于深度学习的推荐系统中，设计了一系列基于非采样学习的高效推荐算法，显著降低了全数据学习的理论时间复杂度，并基于所设计的高效非采样算法框架，分别设计并构建出不同应用场景下的推荐模型，均取得了非常显著的推荐效果[6][7][8][9][10]。

此外，计算机视觉、自然语言处理等领域的最新成果也在不断尝试与推荐系统进行有机结合，例如文献[11][12]。

1.3 本文的工作

本文对基于深度学习的隐反馈 user-based 协同过滤推荐进行了研究，复现了相关文献的前沿算法，进行了丰富的对比实验，并将 sketch-based 的大规模流式数据频繁项统计技术与深度推荐模型混合，提出了基于个性化精排序的实时热门电影推荐方法。

本文对普通电影推荐场景提出了一套完整的系统分析、设计与实现方案。主要设计和实现了用户端和后台管理端，包括电影推荐、信息管理、系统管理等主要业务功能。其中，电影推荐功能包括个性化推荐和热门推荐。对于个性化推荐，本文采用了前沿的算法模型。对于热门推荐，本文采用基于个性化精排序的实时热门电影推荐方法。

本文从功能性、可靠性等方面对电影推荐系统进行测试，结果表明电影推荐系统基本满足了个性化推荐及数据管理需求。

最后对工作进行总结，并对今后推荐算法升级和系统扩展的内容和方向进行展望。

1.4 本文的组织架构

本文共分为五章，每一章的重点内容如下：

第一章，绪论。首先介绍了推荐系统的研究背景和国内外的研究概况。然后对本文的主要工作进行概述。最后简要概述了本文的组织架构。

第二章，相关理论和技术概述。本章首先主要介绍了传统且经典的几种协同过滤算法。接着，介绍了多层感知机和卷积神经网络两种深度学习模型，及系统开发框架，包括 Pytorch、Spring Boot、Vue；中间件 MinIO；运维技术 Docker。另外，还有数据库技术 MySQL 和 MongoDB。最后，简要概述推荐系统评价指标及本文所使用的数据集。

第三章，神经协同过滤方法的研究与实验。本章首先简要介绍矩阵分解的基本原理，并引出广义矩阵分解。其次，详细介绍了神经协同过滤模型框架及卷积神经协同过滤模型框架。然后，介绍非采样的高效神经矩阵分解方法的理论知识和模型实例及本文提出的基于个性化精排序的实时热门物品推荐方法。最后，展示及分析在 MovieLens-1m 数据集上进行的丰富对比实验。

第四章，电影推荐系统的设计与实现。本章按照软件工程标准文档流程，首先进行需求分析，其次

是系统设计，然后是系统的实现及实现效果的展示，最后是系统的测试。本系统以 Spring Boot + Vue 前后端分离开发；相关数据来源于公开数据集 MovieLens-1m 及网络爬虫，运用关系型数据库 MySQL 存取主要电影、用户数据，非关系型数据库 MongoDB 存取次要的历史记录。

第五章，总结与展望。总结本文介绍的协同过滤推荐算法及核心系统开发技术，总结本文所做的工作，并对未来提出展望。

2 相关理论和技术概述

2.1 协同过滤算法概述

协同过滤（Collaborative Filtering，CF）是应用最早且最成功的推荐算法，同时也是当前最流行的推荐算法之一。其本质思想在于，推荐系统不断学习用户行为数据，从而使得推荐列表足够聪明，仅展示用户可能感兴趣的物品，自动过滤用户不感兴趣的物品。

在早期，这种学习过程只是简单的相似度计算或者是数学意义上的矩阵分解。

对于相似度计算，归属于基于邻域的协同推荐。主要分成两类，基于用户的协同过滤算法（UserCF）和基于物品的协同过滤算法（ItemCF）[1]。推荐流程可分为两步：（1）计算用户（物品）相似度。（2）根据用户（物品）相似度和用户的历史行为做出推荐。经典的相似度计算主要采取 Jaccard 公式或余弦相似度实现。公式如下：

$$Jac\_w_{uv}=\frac{|N(u)\cap N(\mathrm{v})|}{|N(u)\cup N(v)|}\quad \mathrm{Cos}\_w_{uv}=\frac{|N(u)\cap N(v)|}{\sqrt{|N(u)||N(v)|}} \tag{1-1}$$

在充分考虑物品的长尾性、用户的活跃度，也有对相似度计算公式的改进。例如 UserCF-IIF、ItemCF-IUF，引入对热门项的权重惩罚[13]。其相似度计算公式如下：

$$w_{uv}=\frac{\sum_{i\in N(u)\cap N(v)}\frac{1}{\log(1+|N(i)|)}}{\sqrt{|N(u)||N(v)|}} \tag{1-2}$$

两者的应用场景也不同，UserCF 以用户群作为推荐衡量的单位，产生偏向于社会化的推荐，粒度较粗，适用于新闻之类的推荐系统。ItemCF 以物品集合作为推荐衡量的单位，产生偏向于个性化的推荐，粒度较细，适用于电商之类的推荐系统。

对于矩阵分解，归属于隐语义模型或可直接称之为矩阵分解模型。其本质思想是利用数学知识，对用户 u、物品 v 交互矩阵 M 进行补全，补全方法需满足使得矩阵特征值变化最小。接着利用补全后的矩阵，进行分解降维，得到用户隐矩阵 P、物品隐矩阵 Q 及对角矩阵 S。最后的评分矩阵计算公式如下：

$$R=PSQ \tag{1-3}$$

直观上，推荐系统也可以看作是稀疏矩阵的补全问题。遗憾的是，纯数学计算补全评分矩阵需要非常大的存储空间且计算复杂度高，只适用于处理大约千维以下的交互矩阵。

Simon Funk 对上述 SVD 矩阵分解模型引入学习机制，采用线性回归思想，避开稀疏问题，以均方差作为损失函数来寻找 P_{u*n}、Q_{v*n} 矩阵，减少了矩阵分解的个数[14]。损失函数公式如下：

$$\mathrm{cost}=\sum(\mathrm{r_{uv}}-\sum_{n-1}^{n}p_{un}q_{vn})+\lambda(\|p_{un}\|^2+\|q_{vn}\|^2) \tag{1-4}$$

迭代公式如下：

$$\frac{\partial C}{\partial p_{un}} = -2q_{vn} \cdot e_{uv} + 2\lambda p_{un} \qquad \frac{\partial C}{\partial q_{vn}} = -2p_{un} \cdot e_{uv} + 2\lambda q_{vn}$$
$$p_{un} = p_{un} + \alpha(q_{vn} \cdot e_{uv} - \lambda p_{un}) \qquad q_{vn} = q_{vn} + \alpha(p_{un} \cdot e_{uv} - \lambda q_{vn}) \tag{1-5}$$

最后的评分矩阵计算公式如下：

$$R = P_{un} Q_{vn}^{T} \tag{1-6}$$

保存训练求得的 P、Q 矩阵，即可按需计算出用户对每个物品的点击率（归一化）。

为了更好得体现用户或物品本身的特性。BiasSVD[15]在 FunkSVD 的基础上引入评分偏差 b_u、b_v，损失函数变为：

$$cost = \sum (r_{uv} - b_u - b_v - \sum_{n-1}^{n} p_{un} q_{vn}) + \lambda_1 (\| p_{un} \|^2 + \| q_{vn} \|^2) + \lambda_2 (\| b_u \|^2 + \| b_v \|^2) \tag{1-7}$$

迭代公式变为：

$$p_{un} = p_{un} + \alpha(q_{vn} \cdot e_{uv} - \lambda_1 p_{un}) \qquad b_u = b_u + \alpha(e_{uv} - \lambda_2 (b_u + b_v))$$
$$q_{vn} = q_{vn} + \alpha(p_{un} \cdot e_{uv} - \lambda_1 q_{vn}) \qquad b_v = b_v + \alpha(e_{uv} - \lambda_2 (b_u + b_v)) \tag{1-8}$$

Koren 将前面提到的基于邻域的方法引入 SVD 模型中，提出 SVD++[16]。在 SVD++中基于邻域的算法是可学习的，以 ItemCF 为例，对于相似度矩阵 R_{ui}' 中 $r_{ui}' = \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} x_i^T y_j$ 且 $cost(w) = \sum (r_{uv} - \sum_{j \in N(u)} (x_i^T y_j) r_{uj})^2 + \lambda (x_i^T y_j)^2$。最后结合 BiasSVD，得 $R = b_u + b_i + Q_{vn}^{T} R_{ui}' + P_{un} Q_{vn}^{T}$。

上述三种矩阵分解算法是最经典的矩阵分解方法。BiasSVD 可以看作 FunkSVD 的改进版本，SVD++可以看作是 BiasSVD 的改进版本。

上述传统的协同过滤方法均存在冷启动问题，即当新用户和新物品加入时，传统协同过滤方法因为没有预先分配交互矩阵相关的行或列，无法产生个性化推荐。另外，UserCF 和 ItemCF 中的相似度度量，由于是数值标量，可能会导致推荐列表物品的排名损失。由于传统矩阵分解中的点积操作是线性的，对于预测评分，仍然可以通过 UserCF 和 ItemCF 中的相似度度量还原。因此，传统的矩阵分解方法同样可能存在排名损失[5]。

现阶段，推荐系统专家将深度学习引入协同过滤，产生了诸多成果。这些模型在一定的泛化性下，能通过定制输入，从而解决冷启动问题。另外，由于引入了非线性因素，深度协同过滤模型能很好的解决排名损失的问题。例如，神经协同过滤(NeuCF)[5]、卷积神经协同过滤(ConvNCF)[17]、Wide&Deep 模型[18]、Deep&Cross 模型[19]等等。

2.2 深度学习模型介绍

2.2.1 多层感知机

深度神经网络是深度学习的基础，其有时候也被称为多层感知机（Multi-Layer perceptron, MLP），属于后向传播学习的前馈型神经网络。网络的一般结构如图 2-1 所示。与传统神经网络相比，其隐藏层足够深，输出也可能不止一个，表达能力大幅增强，具有强大的非线性拟合和自适应学习能力，对于一些无需解释成因的学习问题，其提供了灵活且多元的解决途径。

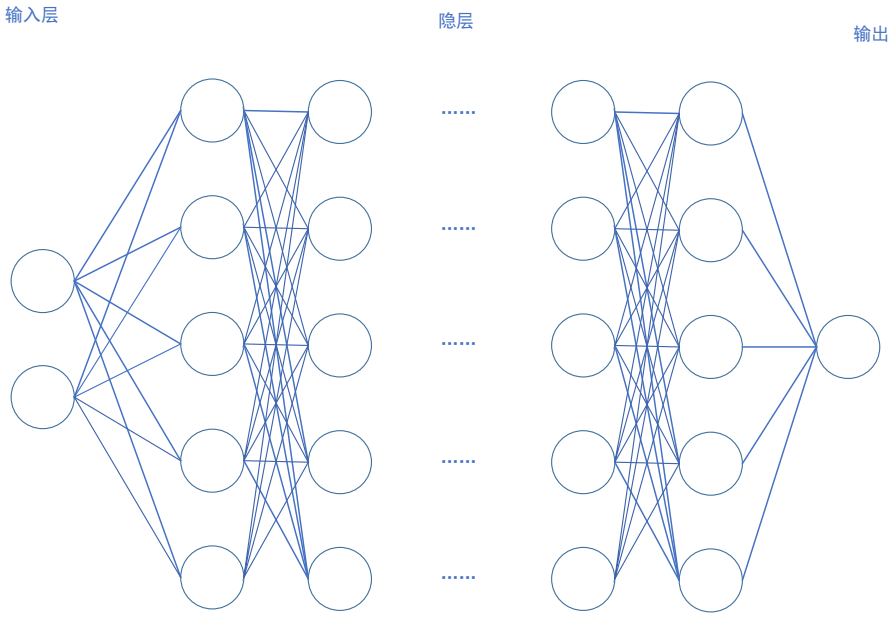

图 2-1　深度神经网络

对于深度神经网络的训练，由于隐层数目多，在反向传播时，容易出现梯度消失和梯度弥散的现象，导致网络失效。对此，常用 Relu 函数代替 Sigmoid 作为激活函数；引入 Batch normalization[20]等方法来解决或缓解梯度消失和梯度弥散。

2.2.2 卷积神经网络

卷积神经网络（Convolutional Neural Networks, CNN）被广泛应用于目标分类、目标检测、图像处理等领域[21]。一个完整的 CNN 可包括卷积层、池化层、全连接层，如图 2-2。卷积和池化相当于特征提取器，而全连接层相当于分类器。

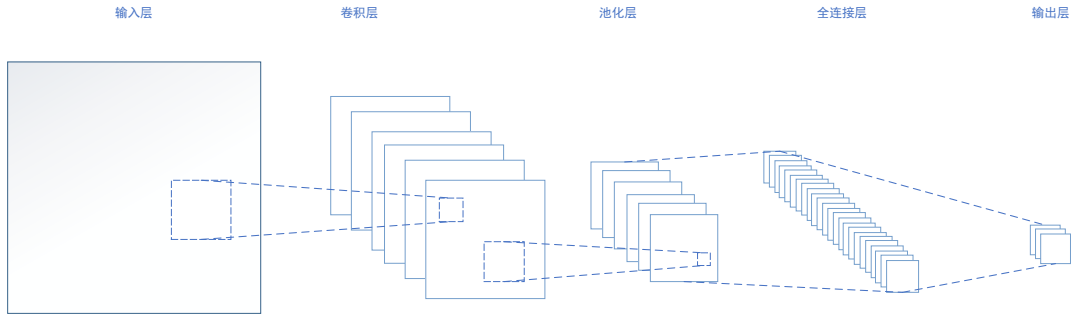

图 2-2　卷积神经网络

卷积层中的卷积操作依赖于卷积核的移动，移动步幅越大，特征映射密度越小。卷积计算中，常用 Relu 函数作非线性激活，以避免梯度消失和梯度弥散。

池化层的目的是降维，但保留了重要信息。池化可以有很多形式，如求和、最大、平均等。

全连接层的目的在于对卷积、池化层所提取的高级特征进行再学习，从而更好得学习特征的非线性组合。在全连接层之前，通常对输入进行 flatten 操作。对于多分类任务，在全连接输出层，根据常用 softmax 作为激活函数。另外，随着模型的复杂，常使用 L1、L2 正则化、Dropout[22]来防止过拟合的发生。

2.3 开发框架、技术与中间件

2.3.1 Pytorch

Pytorch 是当前最流行的深度学习框架之一。它由 Facebook 人工智能研究院于 2017 年推出。近年来，其在学术界的热度已经逐渐超过 TensorFlow。

Pytorch 的核心功能封装于其中的 torch、torch.autograd、torch.nn 和 torch.optim 包中。torch 包提供数据支持，可以看作是一个像 NumPy 一样的 Tensor 库，不过其具有 GPU 支持，即能进行 cuda 运算。torch.autograd 包提供自动微分功能，可以支持 torch 中的所有可微分的 tensor 操作。torch.nn 中包含了与 autograd 深度集成的神经网络库，使用非常灵活。torch.optim 则为 nn 相关的优化包，包含 SGD、Adagard、Adam 等常用优化技术[23]。

Pytorch 运行速度快、内存使用效率高。对于构建模型，上述四个包的使用必不可少，但并不复杂，因其提供了直观的线性构建方式。构建好的模型易于阅读，且能够即时执行、透明调试，具备一定扩展性，极大提高了研发效率。

2.3.2 Spring Boot

Spring Boot 是由 Pivotal 团队研发的一个开源 Spring 轻量级框架，于 2014 年正式发布。它继承了 Spring4.0 的优秀特性，并改进和优化了缺点。其中包含两个重要策略：开箱即用策略和约定优于配置策略。开箱即用的策略通过将开发人员从复杂的配置和依赖项管理中解放出来，帮助他们更多地关注业务逻辑。约定优于配置策略以少量灵活性作为牺牲，减少了开发人员的框架配置，极大提高了开发效率。

2.3.3 Vue.js

Vue 是目前最流行的一套渐进式框架，其用以构建用户交互界面。Vue 中的组件是自底向上渲染的。它仅专注于视图层，易于使用。另外，Vue 还能与第三方的库整合。使用 Vue 脚手架独立开发前端，已成为目前 Web 应用开发的趋势。

2.3.4 MinIO

MinIO 是一个基于 Apache License V2.0 开源许可的轻量级对象存储中间件服务，拥有海量、安全、低成本、高可靠的特点，适用于大容量非结构化数据的存储，例如图片、视频等，并支持单机和分布式两种部署方式，其中后者还提供了纠删码功能，以降低数据丢失风险。同时，MinIO 能简单地与其他应用结合，是一个兼具性能和轻量的中间件。

2.3.5 Docker

Docker 是一个开源的应用容器引擎，是当前最热门的运维技术之一。Docker 中实例化的容器可以看作是具有隔离功能的轻量虚拟机。网络配置分为 bridge（默认）、host、none 三种，相对简单。运用 docker 可以自动化打包和部署任何应用，极大减轻了二次环境配置的痛苦。

2.4 数据库

2.4.1 MySQL 数据库

MySQL 是瑞典 MySQL AB 公司发布的一款开源、跨平台、高效的关系型数据库系统，广泛应用于中小型网站或应用程序中。MySQL 功能强大，支持多种开发语言，如 C、C++、Java、Python 等等。另外，其提供多种数据存储引擎，支持强大的内置函数、视图、事务处理、存储过程和触发器等，安全性高、运行速度快。

2.4.2 MongoDB 数据库

MongoDB 最早由 MongoDB Inc 公司于 2007 年研发，是一款 NoSQL 非关系型文档数据库。所谓 NoSQL 指的并不是 No SQL，而是 Not Only SQL，没有固定的表结构，没有严格遵循 ACID 特性，通常不存在表连接操作，但具备灵活的表级水平扩展性，支持海量数据的存储。文档型数据库则主要用于存储和检索文档数据，可以根据内容和类型进行自我描述，记录的数据远比关系型数据库复杂，适用于不需要考虑关系及约束的场景。MongoDB 支持可以存储复杂数据类型的 bson 格式数据结构，且能按需建立查询索引，极易部署和使用。

2.5 推荐系统评价指标

推荐系统的好坏由多方衡量。一是研发人员，研发人员对推荐系统的评价往往采用离线实验来进行。二是用户，体现于用户对推荐的满意程度。由于离线实验指标和商业指标存在一定差异，由此，系统上线前进行 AB 测试或系统上线后进行用户调查至关重要。推荐系统的主要评价指标可以有：用户满意度、召回率、精度、覆盖率、新颖性、惊喜度、信任度、实时性、健壮性、商业目标[1]。其中召回率、精度、覆盖率这三个指标常用与离线实验。公式如下：

$$\begin{aligned} Recall &= \frac{\sum_{u\in U}|R(u)\cap T(u)|}{\sum_{u\in U}|T(u)|} \\ \text{Precision} &= \frac{\sum_{u\in U}|R(u)\cap T(u)|}{\sum_{u\in U}|R(u)|} \\ Coverage &= \frac{|\cup_{u\in U}R(u)|}{|I|} \end{aligned} \tag{1-9}$$

其中，U 表示所有用户集合，I 表示所有物品集合。R(u)为用户 u 的推荐列表，T(u)表示用户 u 的真正交互物品集。另外，还有两种常用离线实验的评价指标 HR@K 及 NDCG@K。

$$\begin{aligned} HR@K &= \frac{1}{|U|}\sum_{u}1(|\mathrm{R}(u)\cap T_u|) \\ NDCG@K &= \frac{1}{Z|U|}\sum_{u}\sum_{i=1}^{K}\frac{2^{1(|\{R(u)^i\}\cap T_u|)}-1}{\log 2(i+1)} \end{aligned} \tag{1-10}$$

其中，K 为推荐列表长度，1(x)为指示函数，当 x>0 时返回 1，否则返回 0。 $R(u)$ 为用户 u 的按推荐质量倒序排列的推荐列表，其中的第 i 个物品为 $R(u)^i$ 。HR@K 是召回率的另一种表达形式，HDCG@K 则反映了推荐列表的排序质量。

2.6 推荐系统数据集

本文实验采用由 GroupLens 提供的公开电影评分数据集 MovieLens-1m[24]。该数据集包含了 2000 年加入 MovieLens 的 6,040 名用户对大约 3,900 部电影的 1,000,209 个匿名评分，交互矩阵密度约为 3.89%。数据格式为四元组（用户 ID、电影 ID、评分、时间戳）。在研究部分，本文仅关注隐式反馈，即用户与物品交互了，则标签值为 1，否则为 0。另外，由于该数据集与 IMDB 电影网存在一定关联性，在系统实现部分，系统数据沿用了此数据集，通过网络爬虫，补全且丰富了电影数据。

2.7 本章小结

本章主要介绍了相关理论和技术概述，包括协同过滤算法，以及本文使用到的深度学习模型：多层感知机和卷积神经网络。另外，介绍了本系统实现所使用到的系统开发框架，包括 Pytorch、Spring Boot、Vue，对象存储中间件 MinIO，数据库技术 MySQL 和 MongoDB，运维技术 Docker，以及常用的推荐系统性能评价指标和本文使用到的数据集。

3 神经协同过滤方法的研究与实验

3.1 广义矩阵分解模型

在矩阵分解中，关注交互矩阵缺失元素补全的通式，文献[5]将常规矩阵分解模型泛化，提出广义矩阵分解模型（GMF），如图 3-1。对于每个缺失元素的预测值有：

$$\hat{y}_{ui} = f(w^T(p_u \odot q_i)) \tag{2-1}$$

其中 f 为激活函数，w 为权值。在常规矩阵分解中，我们所学习的是 p_u 和 q_i，而 w 则是全为 1 的定值，且激活函数可以看作为 $f(x)=x$。在广义矩阵分解中，w 是从数据中学习的，且激活函数 f 可以是非线性的。显然，广义矩阵分解具有更好的表达能力。

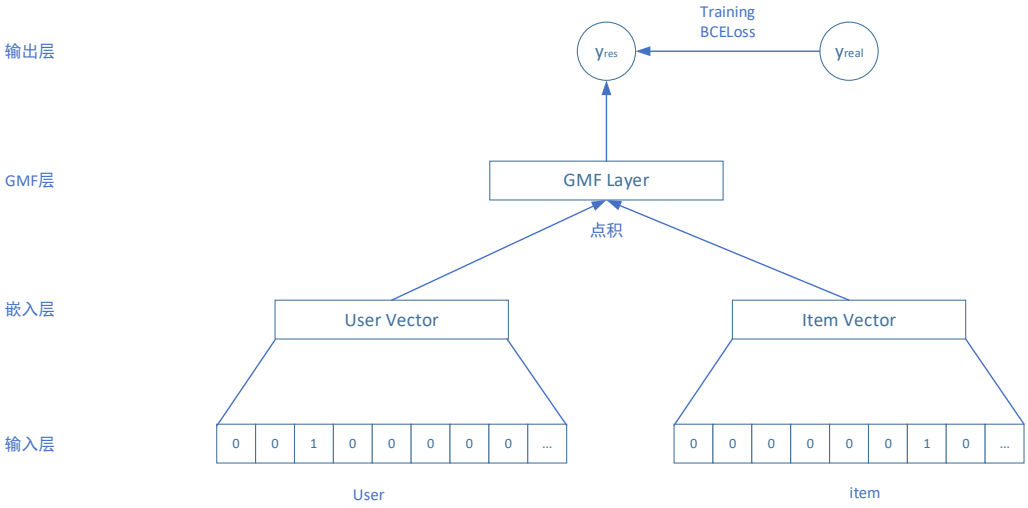

图 3-1　广义矩阵分解推荐模型

3.2 负采样基础神经协同过滤模型

文献[5]在提出广义矩阵分解的基础上，进一步研究，引入塔式结构的多层感知机（非线性），提出神经协同过滤模型（NeuCF），如图 3-2。

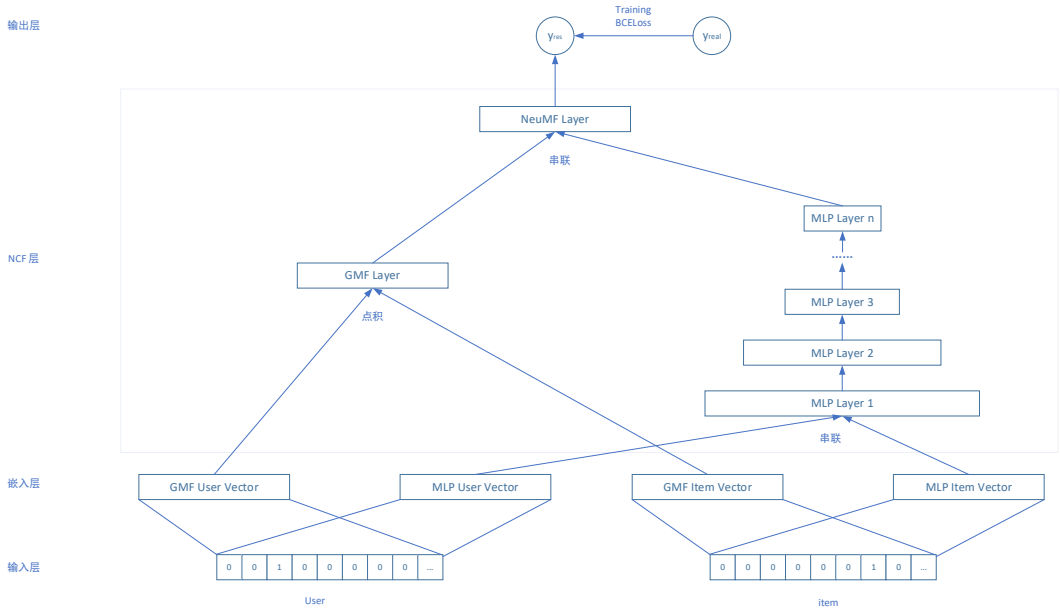

图 3-2　神经协同过滤推荐模型

该模型从横向及纵向加强了常规矩阵分解的表达能力。横向的混合采用简单的连接操作。对于每个缺失元素的预测值有：

$$\hat{y}_{ui} = f(W_{out}^{T}\begin{bmatrix}\overbrace{p_u^{GMF}\odot q_i^{GMF}}^{GMF\ Part}\\ \underbrace{\Phi_n(W_{MLP-n}^{T}(\Phi_{n-1}(...\Phi_2(W_{MLP-2}^{T}[\begin{matrix}p_u^{M}\\ q_i^{M}\end{matrix}]+b_2)...))+b_n)}_{MLP\ Part}\end{bmatrix}) \tag{2-2}$$

其中 Φ_n 为 MLP 层的第 n 层的激活函数，W_{MLP-n} 为对应层权值。f 为预测层的激活函数。W_{out} 表示其权值。

对于隐反馈数据集，为了方便消除小数乘法的下溢出问题，损失函数选择二元交叉熵损失：$cost = -\sum_{(u,i)\in D} y_{ui}\log\hat{y}_{ui} + (1-y_{ui})\log(1-\hat{y}_{ui})$。每一条训练数据由（用户 u 的 one-hot 编码, 物品 i 的 one-hot 编码, 标签）三部分组成。训练集的构成采用负采样的方式：对每个观测正例，随机采取 n 个未观测实例作为负例，在正负样本平衡方面（n 的取值），作者进行了广泛的实验，实验结果表明，模型的性能是对负采样十分敏感的，且对于不同的数据集，没有统一的负采样标准，不过最好的是每个用户的正负样本均衡且用较大概率对热门却没有行为的物品采样。

神经协同过滤模型天然优美，有机的混合了传统矩阵分解与深度神经网络，训练样例少，速度快，并且在粗调参数即能到达较好的实验效果。遗憾的是，其采用了负采样的方式来构建数据集，没有充分利用数据，不具备较强的鲁棒性，且在模型横向对中间结构采取连接操作，只保留了原始的信息。

3.3 负采样卷积神经协同过滤模型

在神经协同过滤中，在用户和物品的嵌入层，采取简单的连接操作，不建模任何关联。为了充分利用不同嵌入维度的相关性，即用户（物品）隐向量维度间的相关性，用户与物品隐向量之间的相关性。在观影方面，直观的例子是，用户年龄段与学历的相关性，电影类型与票房之间的相关性，用户年龄段与观影类别的相关性。在隐向量中，各维度是抽象的。对此，文献[17]提出卷积神经协同过滤模型（ConvNCF），如图 3-3。

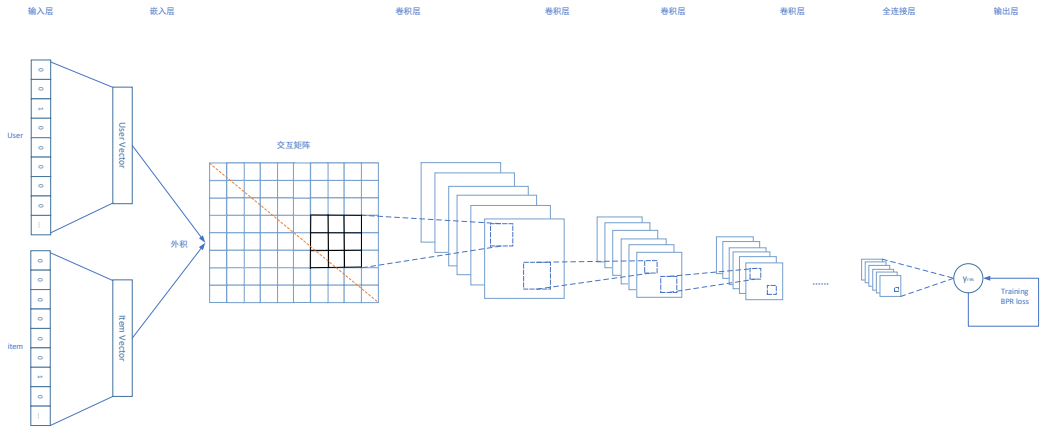

图 3-3　卷积神经协同过滤推荐模型

该模型在嵌入层之后使用外积替代简单的连接操作。注意，外积的结果矩阵的对角线，等于矩阵分解的点积，即该模型仍然包含广义的矩阵分解模型，但具备更多的信息量。对嵌入层的处理结果，由于是二维矩阵，作者采用卷积神经网络来对其进行高级特征提取，并在输出层做一次全连接来进行结果预测。

对于隐反馈数据集，考虑到正例的预测值应比未观测实例的预测值大这一规律。卷积神经协同过滤采用 BPR[25]作为损失函数，公式如下：

$$cost = \sum_{(u,i,j)\in D} -\ln\sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \text{Regularization item} \tag{2-3}$$

其中，$D=\left\{(u,i,j)\middle|i\in y_u^+\wedge j\notin y_u^+\right\}$，$y_u^+$ 表示已观测到的正例。每一条训练数据由（用户 u 的 one-hot 编码，正例物品 i 的 one-hot 编码，未观测物品 j 的 one-hot 编码）三部分组成。训练集的构建同样是采用了负采样的方式。

卷积神经网络卷积核通常不会太大，这导致 ConvNCF 的参数远远比 NCF 少得多。在设计上，ConvNCF 是要优于 NCF，因其额外考虑了嵌入层间的相关性且参数远少于 NCF。不过，其仍然采用了负采样的方式来构建数据集，不具备较强的鲁棒性，且对正则化工作要求较高。

3.4 非采样高效神经矩阵分解

由于文献[26]中提出的非采样加权损失函数 $\text{cost}=\sum_{u\in B}\sum_{v\in V}w_{uv}(y_{uv}-\hat{y}_{uv})^2$（其中 w_{uv} 表示 y_{uv} 的权值）的计算复杂度为 $O(|B||V|d)$（其中 B 表示一批次的用户样本数，V 表示物品集合，d 表示 d 为隐向量间的点积操作），在现实数据集中，V 可能动辄是千万级乃至更高的，如上的理论复杂度，是不能接受的。张敏等人对上述损失函数进行分析推导，分析推导过程如下：

$$
\begin{aligned}
\text{cost} &= \sum_{u\in B}\sum_{v\in V} w_{uv}(y_{uv}-\hat{y}_{uv})^2 \\
&= \underbrace{\sum_{u\in B}\sum_{v\in V} w_{uv}\hat{y}_{uv}^2}_{\downarrow} - 2\sum_{u\in B}\sum_{v\in V^+} w_{uv}^+\hat{y}_{uv} \qquad \left(\text{代入}y^+\equiv 1\text{和}y^-\equiv 0\text{并舍弃常数项}\right) \\
&= \sum_{u\in B}\sum_{v\in V^+} w_{uv}^+\hat{y}_{uv}^2 + \underbrace{\sum_{u\in B}\sum_{v\in V^-} w_{uv}^-\hat{y}_{uv}^2}_{\downarrow} - 2\sum_{u\in B}\sum_{v\in V^+} w_{uv}^+\hat{y}_{uv} \\
&= \underbrace{\sum_{u\in B}\sum_{v\in V^+} w_{uv}^+\hat{y}_{uv}^2 + \sum_{u\in B}\sum_{v\in V} w_{uv}^-\hat{y}_{uv}^2 - \sum_{u\in B}\sum_{v\in V^+} w_{uv}^-\hat{y}_{uv}^2 - 2\sum_{u\in B}\sum_{v\in V^+} w_{uv}^+\hat{y}_{uv}}_{\downarrow} \\
&= \sum_{u\in B}\sum_{v\in V^+}\left(\left(w_{uv}^+-w_{uv}^-\right)\hat{y}_{uv}^2 - 2w_{uv}^+\hat{y}_{uv}\right) + \underbrace{\sum_{u\in B}\sum_{v\in V} w_{uv}^-\hat{y}_{uv}^2}_{\downarrow} \\
&= \sum_{u\in B}\sum_{v\in V^+}\left(\left(w_{uv}^+-w_{uv}^-\right)\hat{y}_{uv}^2 - 2w_{uv}^+\hat{y}_{uv}\right) + \underbrace{\sum_{i=1}^{d}\sum_{j=1}^{d}\left(\left(h_i h_j\right)\left(\sum_{u\in B} p_{u,i}p_{u,j}\right)\left(\sum_{v\in V} w_{uv}^- q_{v,i}q_{v,j}\right)\right)}_{\text{代入}\hat{y}_{uv}^2=\sum_{i=1}^{d}h_i p_{u,i}q_{v,i}\sum_{j=1}^{d}h_j p_{u,j}q_{v,j}=\sum_{i=1}^{d}\sum_{j=1}^{d}\left(h_i h_j\right)\left(p_{u,i}p_{u,j}\right)\left(q_{v,i}q_{v,j}\right)}
\end{aligned}
$$

第一项为正例的损失，由于正例数量一般非常小，该项时间复杂度可忽略不计。第二项中由于 $\sum_{u\in B} p_{u,i}p_{u,j}$ 及 $\sum_{v\in V} w_{uv}^- q_{v,i}q_{v,j}$ 相互独立，它们可以预先计算。由此，时间复杂度从 $O(|B||V|d)$ 显著下降为 $O\left(\left(|B|+|V|\right)d^2\right)$。在此思想下，张敏等人提出了非采样高效神经矩阵分解模型（ENMF）[6]，如图 3-4。每一条训练数据由（用户 u 的 one-hot 编码，所有物品 i 的 one-hot 编码）两部分组成。

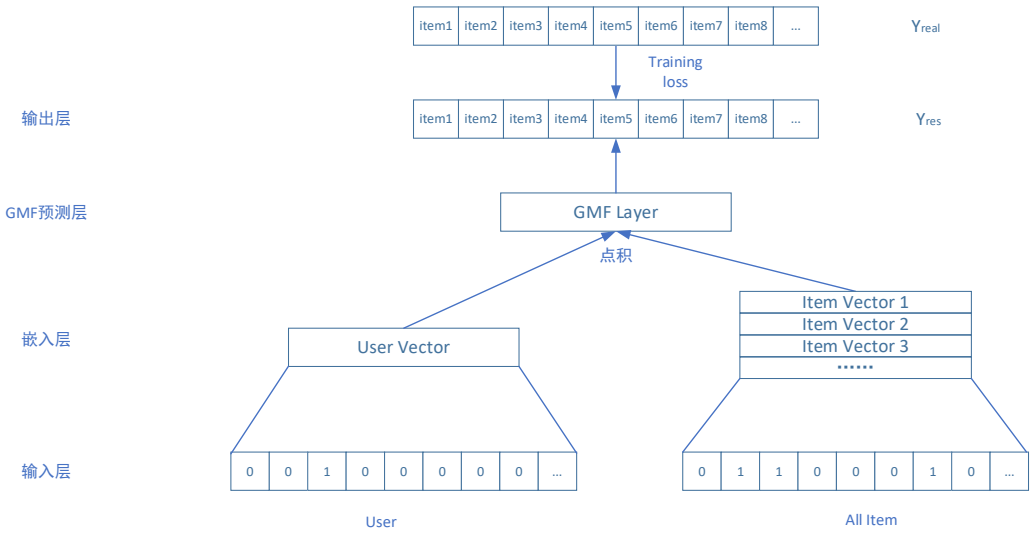

图 3-4　高效非采样神经矩阵分解模型

ENMF 作为非采样方法，训练速度比传统的非采样方法快了 30 倍，并且在实验结果上显著优于上述负采样方法。另外，其参数数量和模型复杂度均少于卷积神经协同过滤模型，但是表现却更好，这反映出了使用非采样学习算法所能带来的训练优势。

3.5 基于个性化精排序的实时热门物品推荐方法

随着用户的访问流，热门项是不断变化的，而统计热门项通常以计数器外加大顶堆实现，当物品数目非常大时，这是非常浪费资源的。

Graham Cormode 等人提出 Count-Min Sketch[27]，在小内存下，利用二维数组，巧妙地解决了大规模流式数据的频度统计问题（如图 3-5），且在理论上能保证误差在可接受的范围。Moses Charikar 等人提出的 Count Sketch[28]同样也能达到不错的效果，但其理论相对复杂，相对难以实现，不过其在非频繁项频度统计方面要优于 Count-Min Sketch。近年来，Lu Tang , Qun Huang 等人提出可逆的 MV-Sketch[29]（如图 3-6），在传统 sketch 算法上，引入 Majority 算法[30]，有效的降低了传统 sketch 中的查询复杂度。

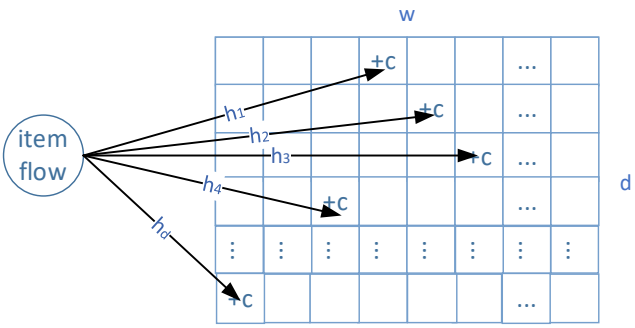

图 3-5　Count-Min sketch 模型

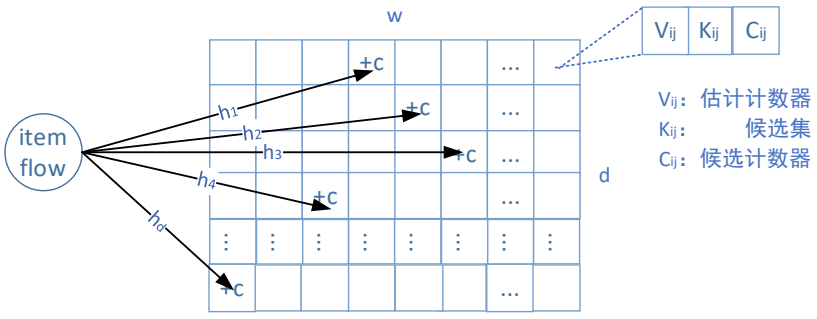

图 3-6　MV-sketch 模型

MV-Sketch 可以说是完成度较高的 sketch 版本，我们可以使用其来对用户访问流的数据进行频度统计，在节省内存的同时，误差仍能保持在可接受的范围，且能迅速进行频繁（热门）项探测。

另外，热门物品推荐常用于解决冷启动问题[1]，但简单的热门推荐，可能不能满足用户的个性化需求。我们考虑先通过 sketch 进行查询出实时热门物品，再使用上文所述的个性推荐算法做热门物品集合的精排序。这样做的好处在于：（1）能够推荐实时热门物品（2）增加了热门物品的信息量（3）提升了用户的使用体验（反映在 NDCG 评价指标）。相关实验将在下一节展示。

3.6 实验设计及结果分析

3.6.1 实验数据集

实验采用 MovieLens-1m 作为数据集。该数据集包含 2000 年加入 MovieLens 的 6,040 名用户对大约 3,900 部电影的 1,000,209 个匿名评分，交互矩阵密度约为 3.89%。数据集划分如下：

（1）验证集。对每位用户随机选取一条正交互作为验证集正例，并选取该用户所有未观测实例。

（2）测试集。对每位用户选取最新的一条正交互作为测试集正例，并选取该用户所有未观测实例。

（3）训练集。选取每位用户不在验证集和测试集的正交互作为训练集正例。对于负采样方法，需对训练集正例进行配套负采样。对于非采样方法，直接选取所有未观测实例。

3.6.2 评价指标

我们选取第二章提到的 HR@K 及 NDCG@K 作为评价指标（公式 2-10）。

$$\begin{aligned} HR@K &= \frac{1}{|U|}\sum_{u} 1(|\mathrm{R}(u) \cap T_u|) \\ NDCG@K &= \frac{1}{Z|U|}\sum_{u}\sum_{i=1}^{K}\frac{2^{1(|\{R(u)^i\}\cap T_u|)}-1}{\log 2(i+1)} \end{aligned} \tag{2-4}$$

其中，K 为推荐列表长度，1(x)为指示函数，当 x>0 时返回 1，否则返回 0。$R(u)$ 为用户 u 的按推荐质量倒序排列的推荐列表，其中的第 i 个物品为 $R(u)^i$ 。两个评价指标均为越高越好。

3.6.3 超参数选取

本文使用 Pytorch 进行实验。对于涉及随机的操作，初始化相同的随机数种子。所有的超参数都根据验证集的评价结果来调节。所有基线方法的参数都按照相应的文献进行初始化，然后仔细调整以达到最佳性能。学习算法的学习率、dropout ratio（不保留比例）、批次大小分别在 [0.005, 0.01, 0.02, 0.05]、[0.1, 0.3, 0.5, 0.7, 0.9]、 [128, 256, 512, 1024]中选取，隐向量维度 d 在[8, 16, 32, 64]中调节，对于每个正例的负采样数量，本文尝试选取了[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]。各参数对算法的影响及算法性能的对比实验将在下一节展示。

3.6.4 实验结果及分析讨论

本文的实验在同一机器上进行，实验环境为 AMD R7-4800H@2.90GHz CPU、一块 NVIDIA RTX 2060 MAX-Q with 6GB 及 16GB 内存。

表 3-1 展示了上述不同模型在 MovieLens-1m 数据集上收敛时的性能。实验结果表示：（1）ENMF 在各个指标上的性能最佳。（2）基于学习的算法，大多比非学习算法（UserCF、ItemCF、ItemPop）表现好。（3）传统协同过滤算法中，UserCF 的表现较优，甚至略优于一些学习算法，但其初始化及产生推荐列表所需的时间要远高于本文研究的深度协同算法，且其存在冷启动问题。（4）ConvNCF 模型最为复杂，却没有较佳的性能，存在质疑。（5）在基于个性化精排序的实时热门电影推荐方法中，ItemPop-NCF 由于模型增加了非线性因素（MLP 层），表现最佳，远远优于 ItemPop。

表 3-1 不同模型在数据集上的性能

	HR@50	HR@100	HR@200	NDCG@50	NDCG@100	NDCG@200
UserCF	0.2752	0.3980	0.5262	0.0840	0.1039	0.1218
ItemCF	0.1651	0.2409	0.3053	0.0438	0.0562	0.0651
ItemPop	0.1828	0.1949	0.2614	0.0372	0.0391	0.0483
ItemPop-NCF	0.2417	0.3705	0.5157	0.0743	0.0951	0.1154
ItemPop-ConvNCF	0.1369	0.2255	0.3086	0.0349	0.0493	0.0610
ItemPop-ENMF	0.1488	0.2853	0.4634	0.0362	0.0582	0.0831
SVD	0.1790	0.2839	0.4267	0.0512	0.0681	0.0880
GMF	0.2571	0.3917	0.5490	0.0778	0.0978	0.1203
NeuCF	0.2402	0.3525	0.5414	0.0728	0.0886	0.1198
ConvNCF	0.0646	0.1440	0.2373	0.0149	0.0278	0.0407
ENMF	0.3005	0.4520	0.6157	0.0888	0.1136	0.1342

在实验设置上，为了公平，嵌入层大小均设置为 64，Dropout ratio 设置为 0.3。图 3-7 展示了不同模型性能随迭代次数的变化。表 3-2 展示了各个达到最佳性能所需的迭代次数、单次迭代耗时及总耗时。由实验结果可知：（1）ENMF 的性能最佳，且到达收敛时总迭代时间最短。（2）在该数据集上，GMF 性能要略优于 NCF。（3）ConvNCF 在该数据集上表现性能曲线表现震荡，不能很好的学习该数据集。（4）由于 NCF 模型具有横向模型扩展运算，导致 NCF 达到收敛时，迭代时间最长。

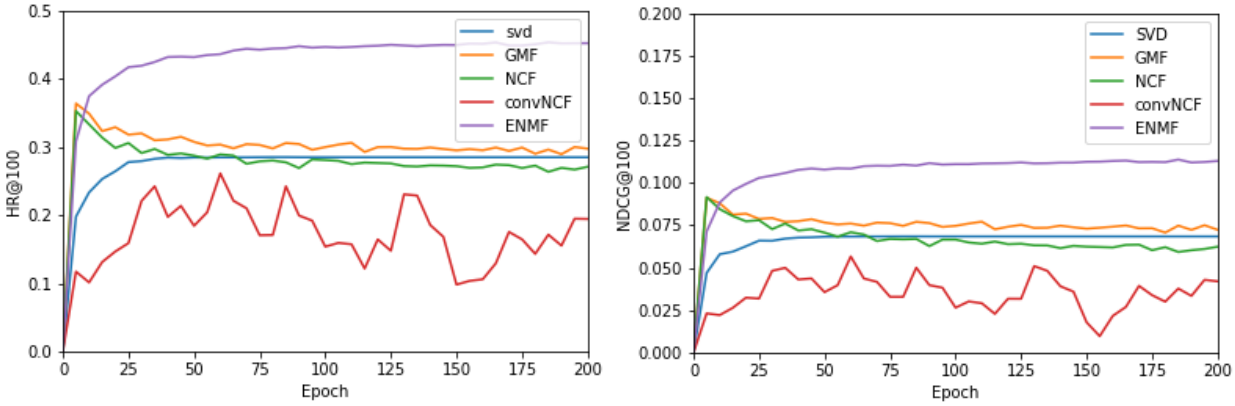

图 3-7 不同模型的性能曲线对比

表 3-2 不同模型的时间效率对比

	Single(s)	Total(s)	Appr. conver epoch
UserCF		1616680893.44	
ItemCF		1616681311.36	
SVD	1616772082.51	80838604125.43	50
GMF	118.04	1180.41	10
NeuCF	201.64	2016.45	10
ConvNCF	7.25	580.26	80
ENMF	0.62	61.63	100

图 3-8 展示了嵌入层大小对模型的影响。当嵌入层为 64 时，各模型效果最佳。除 ConvNCF 外，其余模型的性能随嵌入层增大而不断提升。

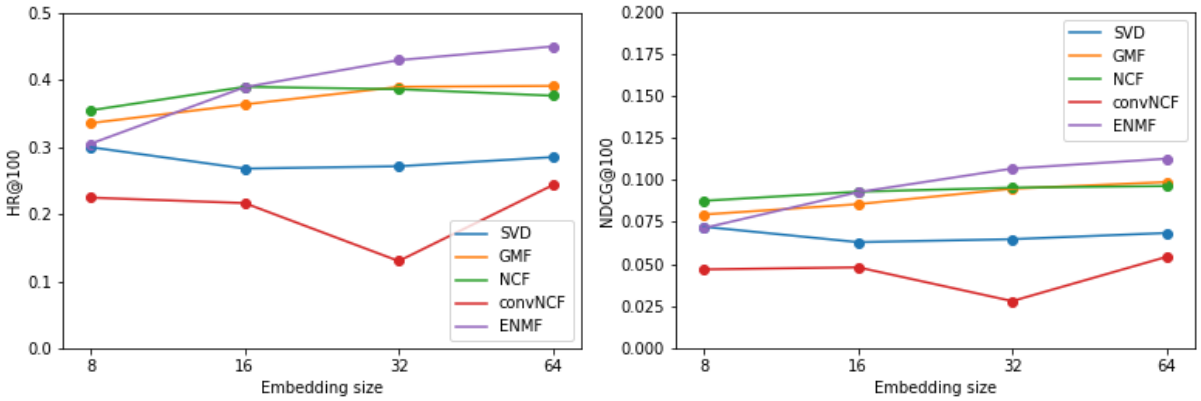

图 3-8 不同模型在不同嵌入层大小的性能对比

图 3-9 展示了负采样模型性能随负采样数量的变化趋势。由实验结果可知，模型在该数据集上，对负采样十分敏感，没有统一的负采样规律。从整体上看，较为可靠的负采样取值为 4，本文的其他部分实验均使用 4 作为负采样数量。

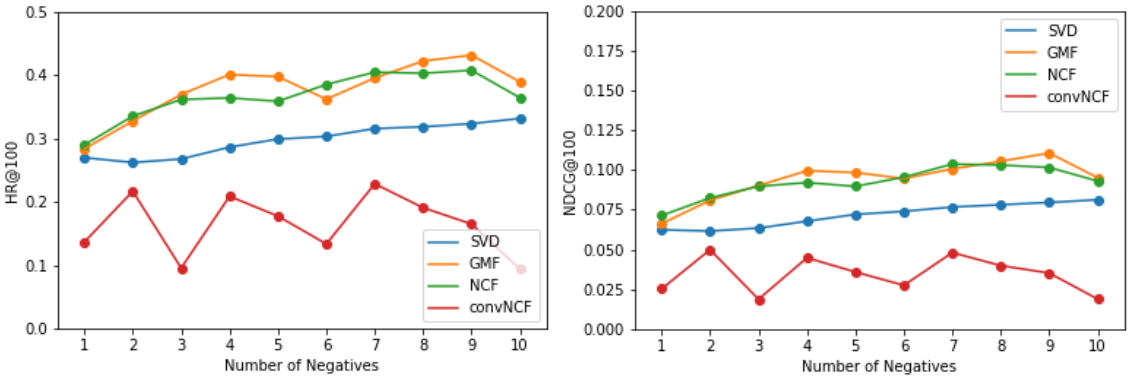

图 3-9 负采样模型性能随负采样数量的变化

图 3-10 展示了 ConvNCF 和 ENMF 模型性能随 dropout ratio 的变化趋势。由实验结果可知，模型在该数据集上，ConvNCF 的最佳 dropout ratio 为 0.7。ENMF 的最佳 dropout ratio 为 0.3，往后当 dropout ratio 增大时，ENMF 的性能表现下降。

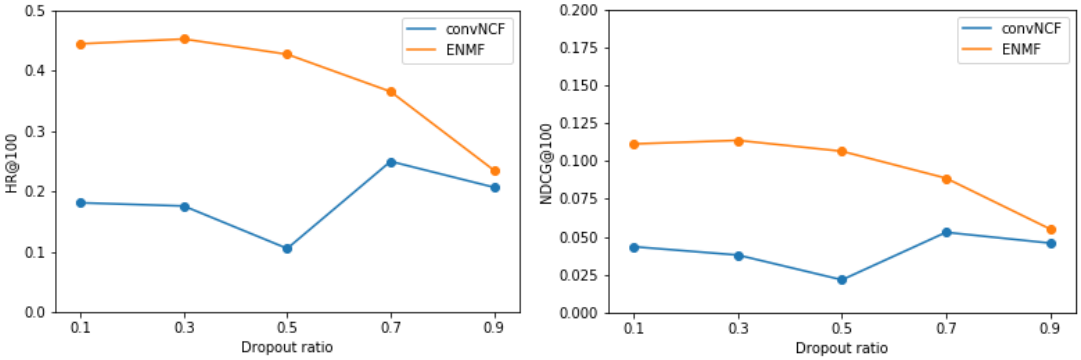

图 3-10 ConvNCF、ENMF 在不同 Dropout ratio 上的性能对比

图 3-11 展示了 ENMF 在不同负权值上的性能对比。当取值 0.1 时，模型已能达到较好的性能，大于 0.1 时，性能变化不大。

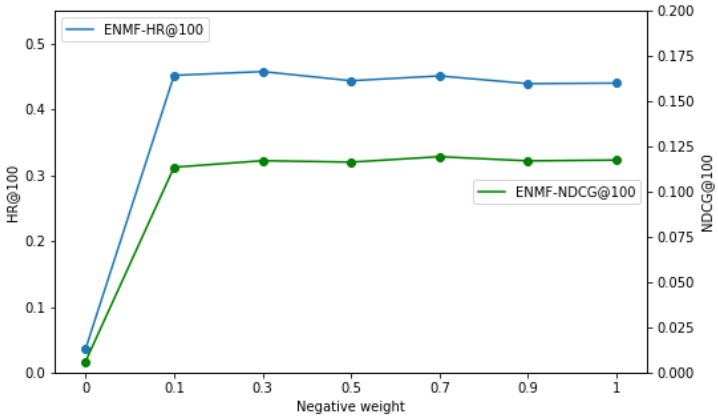

图 3-11　ENMF 在不同负权值上的性能对比

3.7 本章小结

本章对深度协同推荐进行了研究。在介绍完数据集划分、评价指标及超参选取后，进行了丰富的对比实验和各个模型参数对模型性能的影响实验。实验结果表明，非采样的 ENMF 具有最优的推荐性能及时间效能。在研究了大数据时代热门推荐所存在的问题后，提出了一种基于个性化精排序的实时热门推荐方法，该方法将 sketch-based 的大规模流式数据频繁项统计技术与深度推荐模型混合。实验结果表明，MV-sketch 与 NCF 模型的混合实例，达到了高效、准确的推荐性能。

4 电影推荐系统的设计与实现

4.1 需求分析

4.1.1 功能性需求

本系统按功能性划分可分为用户端和后台管理端两部分。

（1）用户端。用户端的主要使用者为网站用户和游客。用户注册后即可访问网站，并浏览电影信息，对电影进行评论。电影信息分为推荐和影视库两部分，用户能根据关键字检索电影。另外，用户在网站内，能自由修改个人信息及密码，查看历史的浏览记录及系统公告。

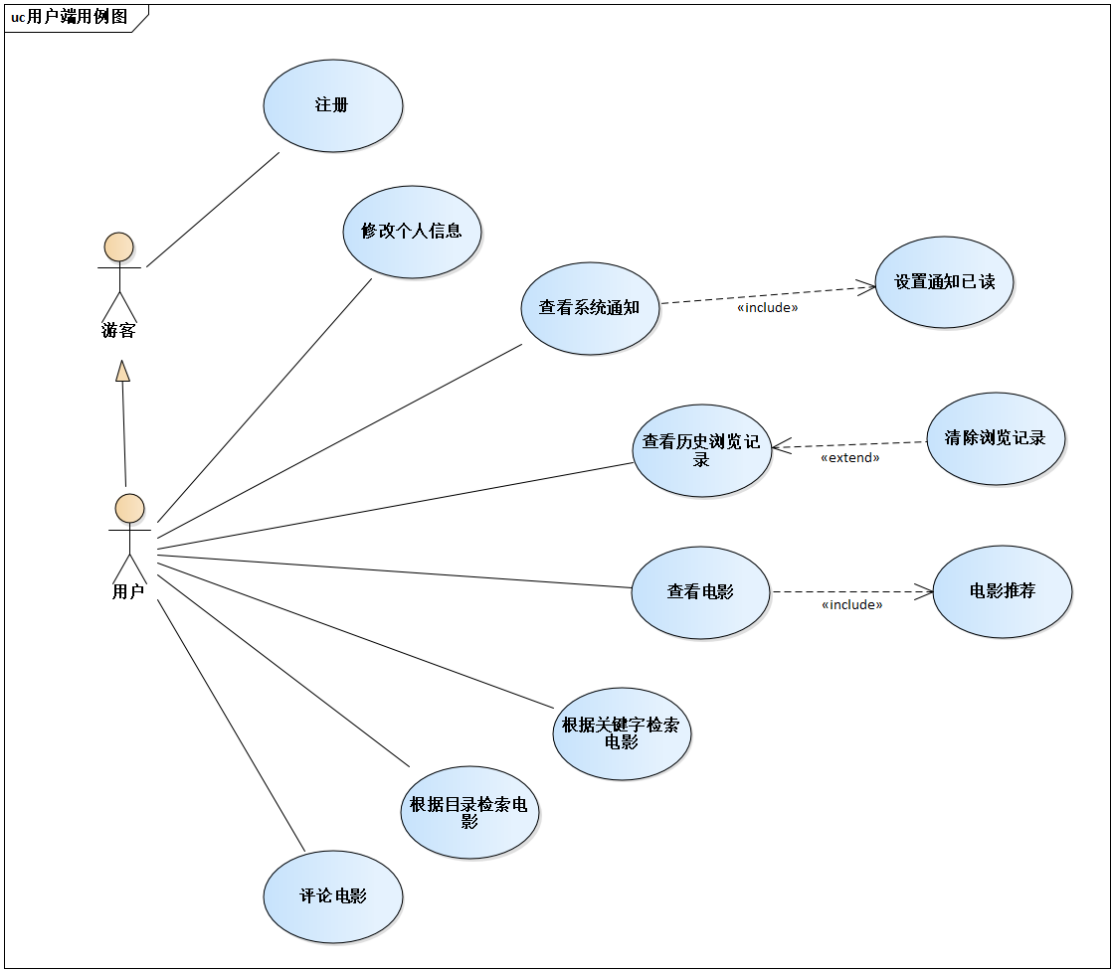

图 4-1 用户端用例图

表 4-1 用户端用例词汇表

编号	用例名称	用例描述	参与者
A001	注册	该用例描述游客在站点填写个人信息（包括用户名、密码、性别、年龄段、职业类型、邮政编码、头像等），进行站点注册，系统将根据用户输入进行合法性验证。注册成功后，游客将成为系统用户，可使用用户名、密码登录网站。	游客
A002	修改个人信息	该用例描述用户登录后，在站点修改个人信息（包括密码、性别、年龄段、职业类型、邮政编码、头像等）。系统根据用户的修改，对用户资料进行更新或显示。	用户
A003	查看系统通知	该用例描述用户查看系统通知。系统根据用户的操作，显示对应的通知详情（包括发出者、发出者头像、发出时间、通知内容）。	用户
A004	设置通知已读	该用例描述用户设置通知已读，包括全部已读及单条通知已读。单条通知已读同时可以通过查看通知详情触发实现。	用户

		系统将根据用户的操作，对通知信息进行更新和显示。	
A005	查看历史浏览记录	该用例描述用户查看历史电影的浏览记录。系统根据用户操作，按时间倒序显示浏览记录（同一电影的新浏览会覆盖旧浏览）。每条浏览记录包括先前浏览电影的电影名、电影海报、电影简介等内容。	用户
A006	清除浏览记录	该用例描述用户查看浏览记录时，可选清除浏览记录，包括全部清除和单条记录清除。系统将根据用户的操作，对浏览记录进行更新和显示。	用户
A007	查看电影	该用例描述用户查看电影信息，包括电影名称、海报、导演、演员、类别、片长、上映时间、简介等。系统将根据用户的操作，对电影信息进行显示。	用户
A008	电影推荐	该用例描述系统向用户推荐电影，分为智能推荐及热门推荐两类。对于智能推荐，推荐模型根据用户建立。系统将根据用户的不同，推荐不同的电影信息。对于热门推荐，系统实时推荐展示当下热门电影基本信息。	用户
A009	根据关键字检索电影	该用例描述用户输入关键字性需求，获取电影。系统根据用户输入的关键字，从电影库中模糊匹配电影名、导演、编剧、简介与关键字相似的电影，并展示给用户。	用户
A010	根据分类目录检索电影	该用例描述用户使用分类目录检索电影（目录包括地区、时间、类别）。系统根据用户操作，对电影信息进行动态显示。	用户
A011	评论电影	该用例描述用户输入文字信息进行电影评论。系统将根据用户的操作，对楼层评论信息进行更新和显示。	用户

（2）后台管理端。后台管理端的主要使用者为管理员。不同权限的管理员能够进行不同的操作，权限的分配与回收由超级管理员通过权限管理功能进行。主要的管理端操作有用户信息维护（包括删除用户、新增用户、修改用户资料、检索用户信息等）；评论管理（包括批量添加评论、修改或删除自己发布的评论等）；电影信息维护（包括删除电影、新增电影、编辑修改电影信息、通过关键字检索电影等）；公告管理（包括批量向用户发布公告、修改或删除公告等）；动态权限管理（包括权限的授予及回收）、查看系统数据报表（如用户职业、年龄、性别分布；活跃用户；热门电影等）；首页轮播图管理（包括轮播图的添加、失效、修改）等。

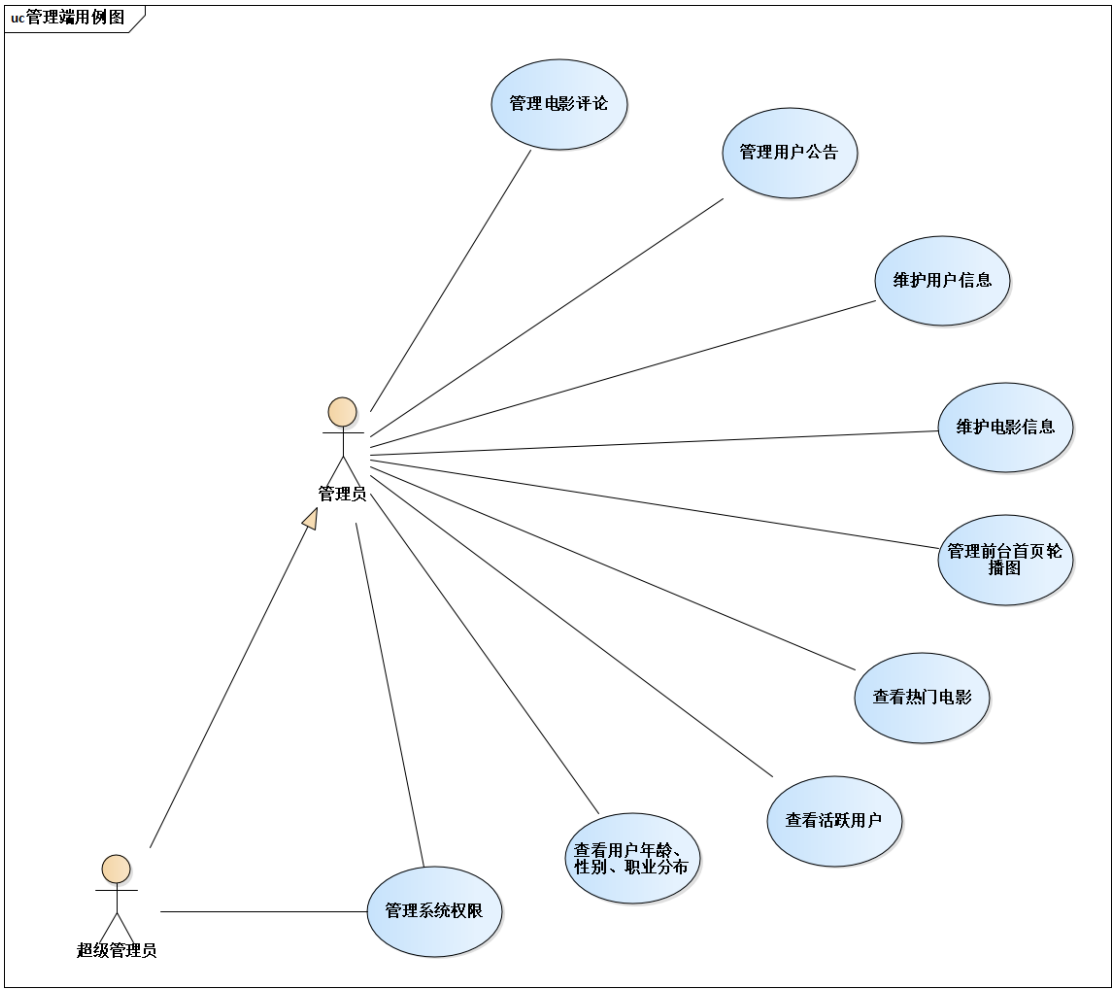

图 4-2 后台管理端用例图

表 4-2 后台管理端用例词汇表

编号	用例名称	用例描述	参与者
B001	管理电影评论	该用例描述管理员删除评论；新增（含批量）评论；修改自己的评论或根据评论人、电影名、评论时间区间查询评论信息。系统根据管理员的操作，对电影评论信息进行更新或显示。	管理员
B002	管理用户公告	该用例描述管理员删除、新增（含批量）、修改公告或根据用户名、管理员名、管理员邮箱和时间区间查询公告信息。系统根据管理员的操作，对用户公告信息进行更新或显示。	管理员
B003	维护用户信息	该用例描述管理员删除、新增、修改用户基本信息（含重置密码）或根据用户名/性别/职业/年龄/邮编的关键字检索用户信息。系统根据管理员的操作，对用户信息进行更新或显示。	管理员
B004	维护电影信息	该用例描述管理员删除、新增、修改电影信息或根据电影名、导演、编剧、演员、类别、发行时间区间查询电影信息。	管理员

		系统根据管理员的操作，对用户信息进行更新或显示。	
B005	管理前台首页轮播图	用户端首页轮播图上限为 4 张。该用例描述管理员对用户端首页轮播图进行新增、删除、更新操作。系统根据管理员的操作，对轮播图进行更新和显示。	管理员
B006	查看热门电影	该用例描述管理员查看实时热门电影。系统根据管理员的操作，实时展示电影库中 Top8 热门电影（包括电影名、点击量）的条形图。	管理员
B007	查看活跃用户	该用例描述管理员查看实时活跃用户。系统根据管理员的操作，实时展示 Top20 热门用户（包括用户名、活跃量）的条形图。	管理员
B008	查看用户年龄、性别、职业分布	该用例描述管理员查看系统用户分析数据。系统根据管理员的操作，展示系统用户的年龄分布柱状图或性别分布饼图或职业分布饼图	管理员
B009	管理系统权限	该用例描述超级管理员维护管理员信息（新增、修改（含重置密码）、删除管理员数据或根据用户名、电子邮箱、权限、创建时间区间查找管理员）；动态赋予或回收管理员的操作权限（包括电影信息维护、用户信息维护、评论管理、公告管理、权限管理、报表及其他）。系统根据超级管理员的操作，对管理员信息进行更新或显示。	超级管理员 管理员

4.1.3 非功能性需求

（1）性能和兼容性需求

为了保证电影推荐系统能够稳定、可靠、高效运行，应该满足以下需求：

①适应性。本系统应体现各类操作系统及 IE8 及以上版本的各类浏览器的兼容性。

②系统处理的及时性和准确性。虽然本系统数据量较大，但要求响应时间应主要受网速影响。在网络通畅的情况下，平均响应时间应在 2 秒以内，且能正确反馈操作结果。在 500 个并发用户的高峰期，该系统内的基本功能，处理能力应至少达到 20TPS，且在高于 2 倍运行压力情况下，能保证全天候地准确、稳定运行。

③易于维护和扩充。电影是人类文化和科技的新产物，人类社会的电影产量蒸蒸日上。设计之初就应该充分考虑日后电影数据信息的维护、更新和扩充，并且预留出动态替换推荐算法的插槽。

（2）安全性需求

为保证电影推荐系统安全性。对于前台用户端，应防止恶意访问，保障用户个人信息安全。对于后台管理端，应进行分权限操作，防止恶意访问、攻击，窃取或恶意修改各类信息数据，以保证操作及数据的安全性，防止有意或无意的破坏行为。如果数据遭到破坏，应能及时恢复。系统应每日自动备份数据，在系统意外崩溃时，造成的损失应能降到最低。

（4）先进性需求

目前，以流行开发框架开发且具备前沿推荐算法的电影推荐系统较为少见。在系统设计上，要求使

用较为成熟和流行的信息系统开发框架。对于算法层设计，要求具备前沿性算法，能做到精准的个性化推荐。

4.2 系统设计

4.2.1 系统体系结构

图 4-3 总体架构图

如图 4-3 所示，整个电影推荐系统大致由四部分组成：页面表现层、信息管理及推荐业务层、推荐算法层、系统数据处理层。其中，推荐算法层的设计将在 4.2.3 小节详细介绍。

依照上一节的功能性需求分析，电影推荐系统的主要业务功能模块，如图 4-4 所示。

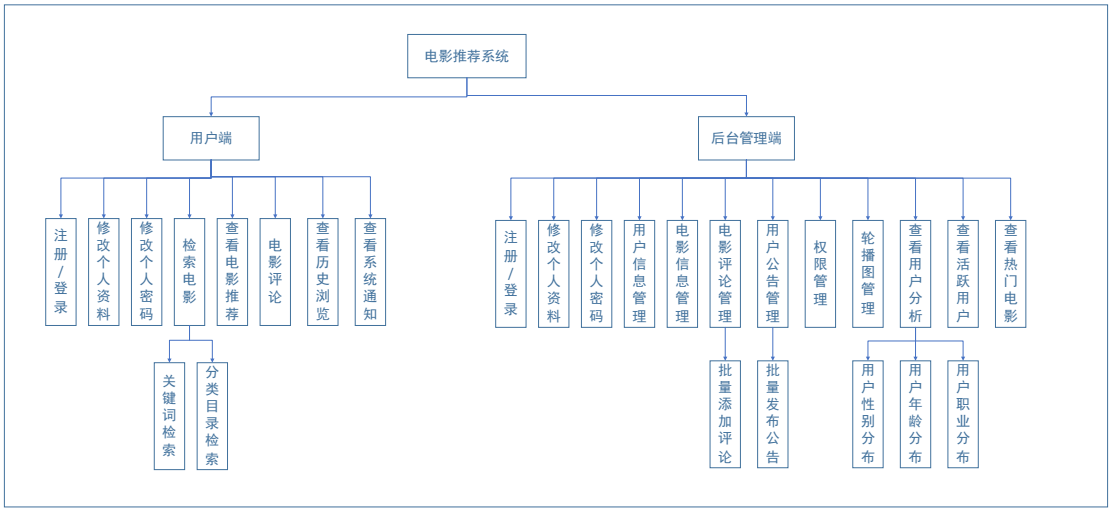

图 4-4 电影推荐系统主要业务功能模块图

4.2.2 交互及结构设计

本小节展示主要用例的顺序图及设计类模型。

(1) 注册（A001）

表 4-3 展示了“注册”用例所包含的接口类、控制类与实体类。其中接口类包括注册页面，控制类包括用户信息管理器，实体类包括用户类。

表 4-3 “注册”用例的接口类、控制类、实体类

接口类	控制类	实体类
注册页面	用户信息管理器	用户

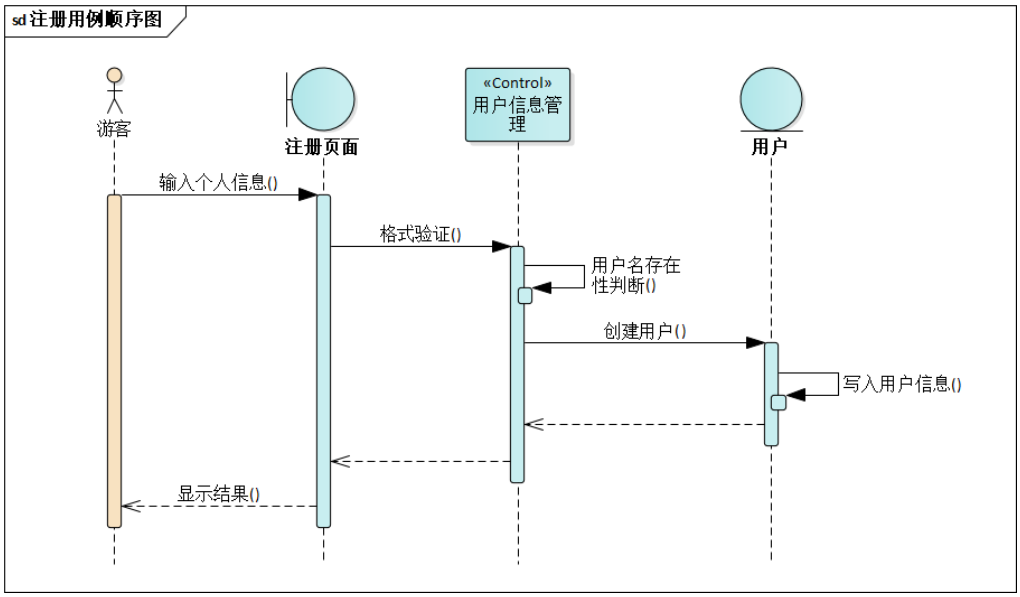

图 4-5 “注册”用例的顺序图

图 4-5 描述了游客注册的前端操作逻辑和后台处理逻辑。“注册”用例中类与类之间的关系图，如图 4-6 所示。

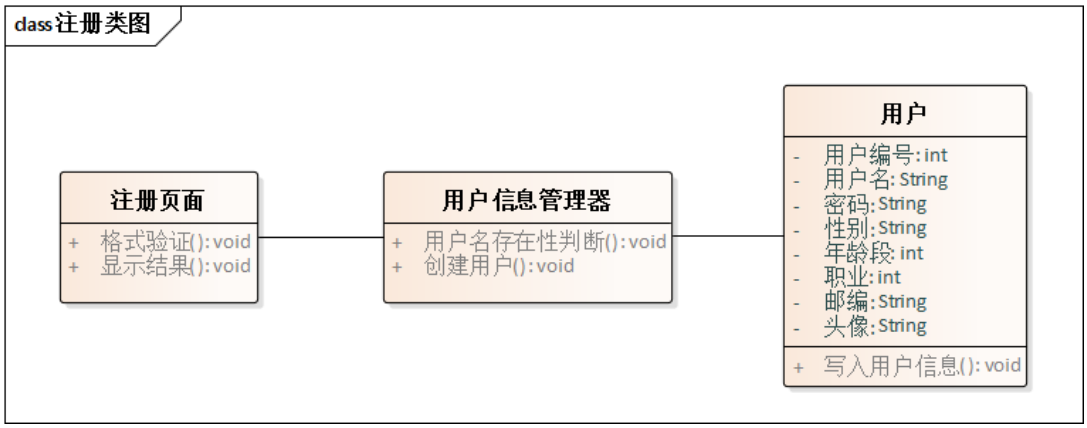

图 4-6 “注册”用例的类图

(2) 查看系统通知（A003）

表 4-4 展示了“查看系统通知”用例所包含的接口类、控制类与实体类。其中接口类包括系统通知页面，控制类包括公告信息管理器、管理员信息管理器与用户通知明细管理器，实体类包括用户通知类、管理员类与通知明细类。

表 4-4 “查看系统通知”用例的接口类、控制类、实体类

接口类	控制类	实体类
系统通知页面	公告信息管理器	用户通知
	管理员信息管理器	管理员
	用户通知明细管理器	通知明细

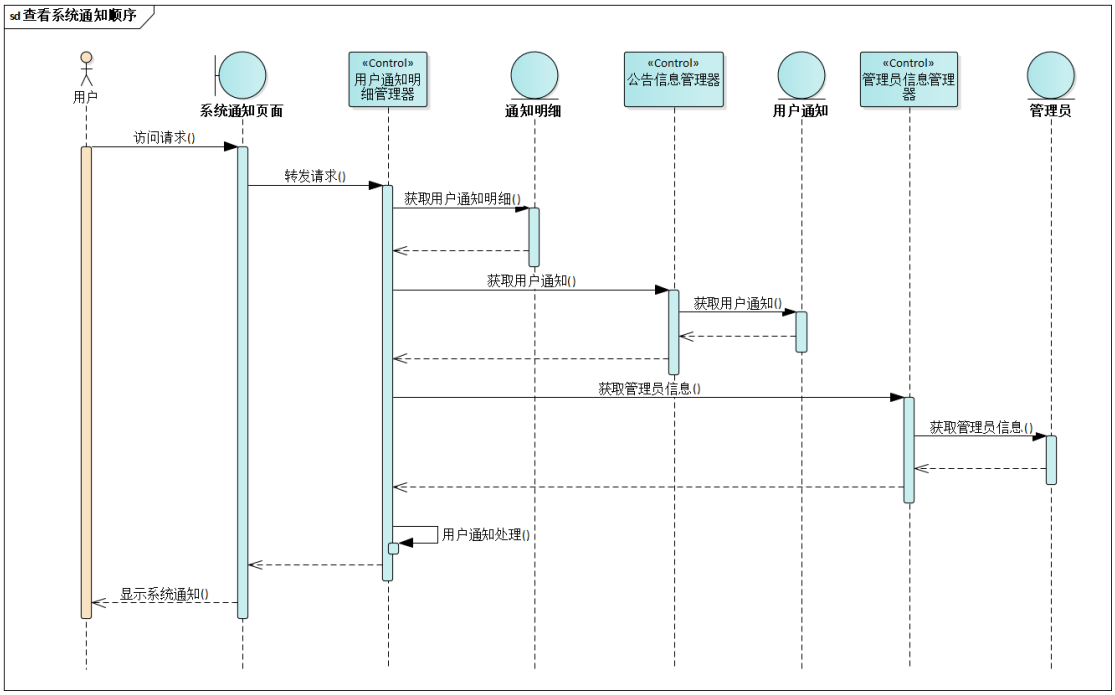

图 4-7 “查看系统通知”用例的顺序图

图 4-7 描述了用户查看系统通知的前端操作逻辑和后台处理逻辑。“查看系统通知”用例类与类之间的关系图，如图 4-8 所示。

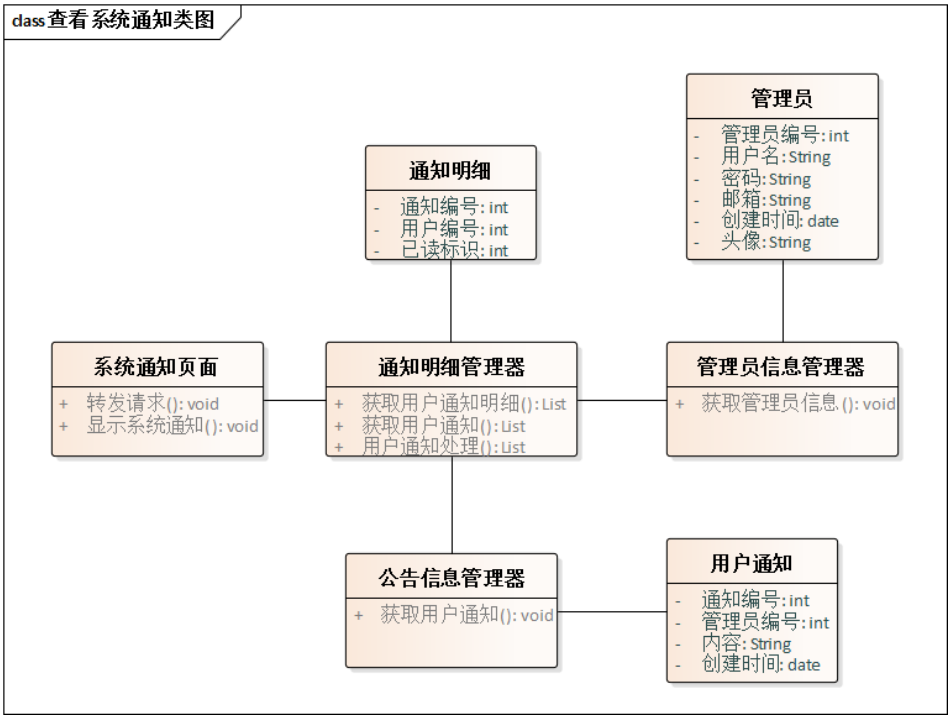

图 4-8 “查看系统通知”用例的类图

(3) 查看历史浏览记录（A005）

表 4-5 展示了“查看历史浏览记录”用例所包含的接口类、控制类与实体类。其中接口类包括历史记录页面，控制类包括记录控制器，实体类包括历史记录类。

表 4-5 “查看历史浏览记录”用例的接口类、控制类、实体类

接口类	控制类	实体类
历史记录页面	记录控制器	历史记录

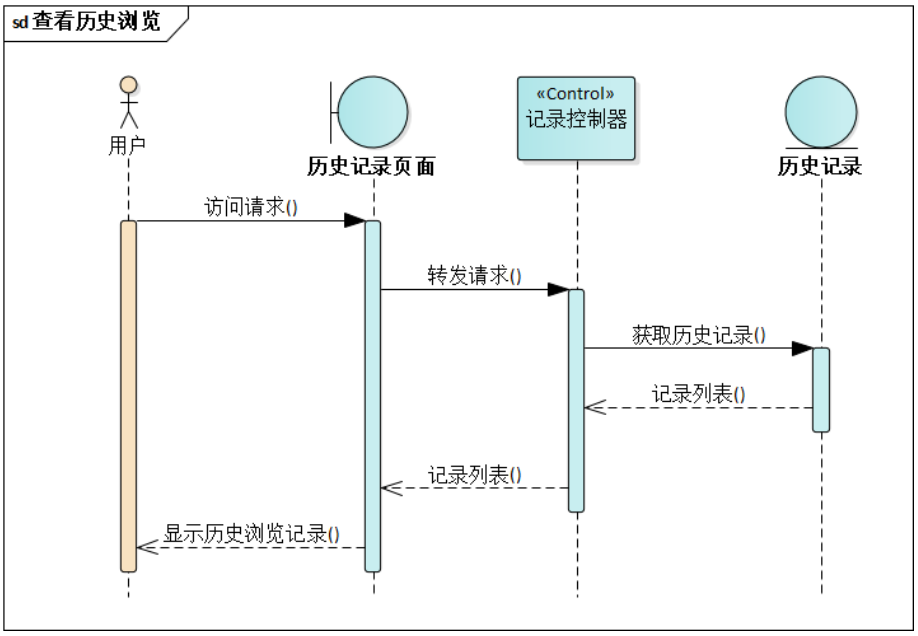

图 4-9 “查看历史浏览记录”用例的顺序图

图 4-9 描述了用户查看历史浏览记录的前端操作逻辑和后台处理逻辑。“查看历史浏览记录”用例中类与类之间的关系图，如图 4-10 所示。

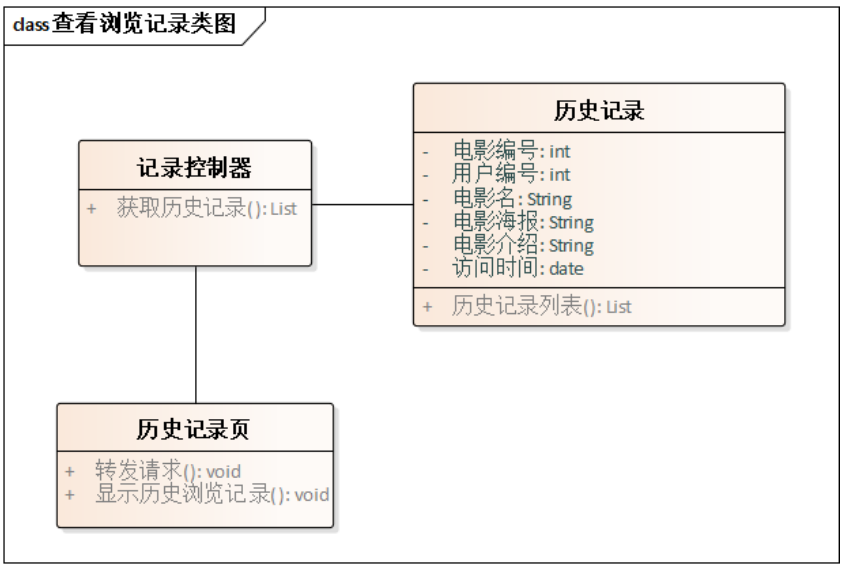

图 4-10 “查看历史浏览记录”用例的类图

(4) 查看电影（A007）

表 4-6 展示了“查看电影”用例所包含的接口类、控制类与实体类。其中接口类包括电影页面，控制类包括电影信息管理器和记录控制器，实体类包括电影类和历史记录类。

表 4-6 “查看电影”用例的接口类、控制类、实体类

接口类	控制类	实体类
电影页面	电影信息管理器	电影
	记录控制器	历史记录

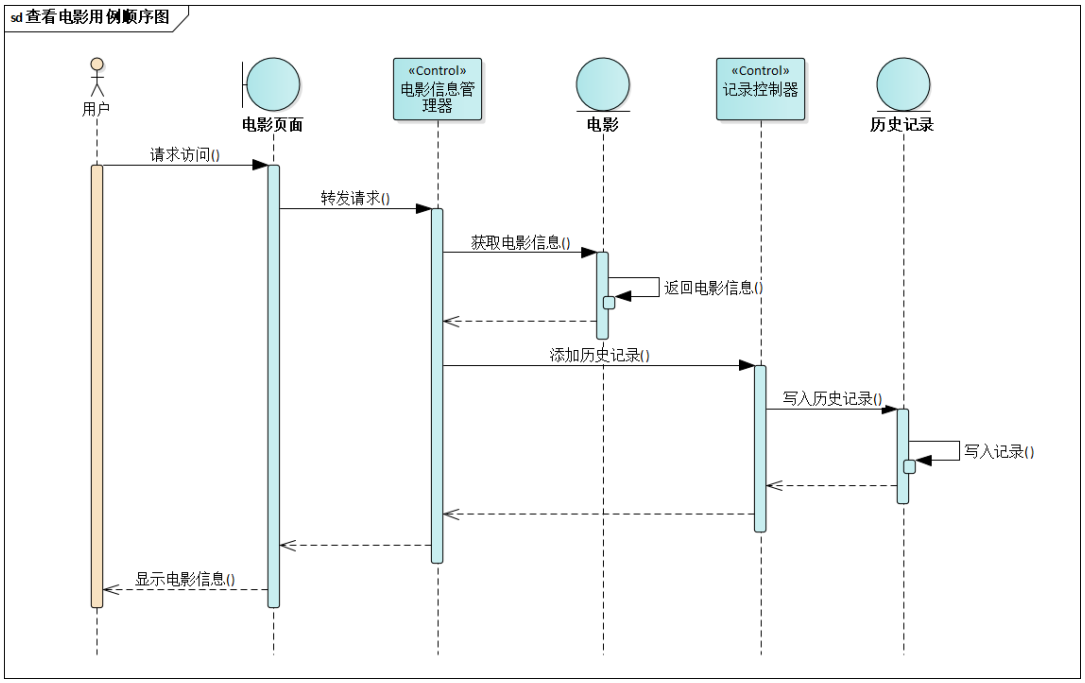

图 4-11 “查看电影”用例的顺序图

图 4-11 描述了用户查看电影的前端操作逻辑和后台处理逻辑。图 4-12 则展示了“查看电影”用例的类图。

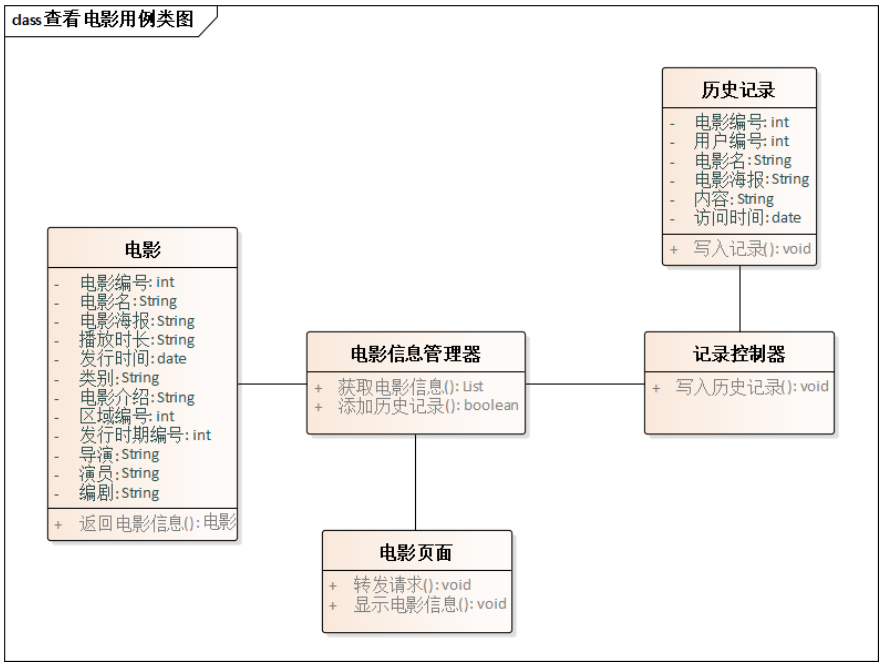

图 4-12 “查看电影”用例的类图

(5) 电影推荐（A008）

表 4-7 展示了“电影推荐”用例所包含的接口类、控制类与实体类。其中接口类包括电影页面，控制类包括电影信息管理器、推荐器，实体类包括电影类。

表 4-7 “电影推荐”用例的接口类、控制类、实体类

接口类	控制类	实体类
电影页面	电影信息管理器	电影
	推荐器	

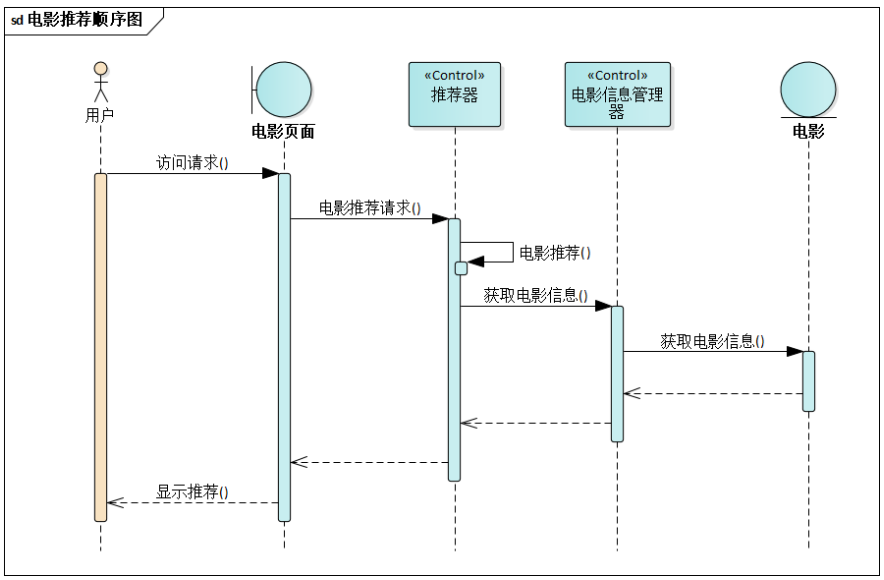

图 4-13 “电影推荐”用例的顺序图

图 4-13 描述了电影页面中电影推荐模块的后台处理逻辑。“电影推荐”用例类与类的关系图，如图 4-14 所示。

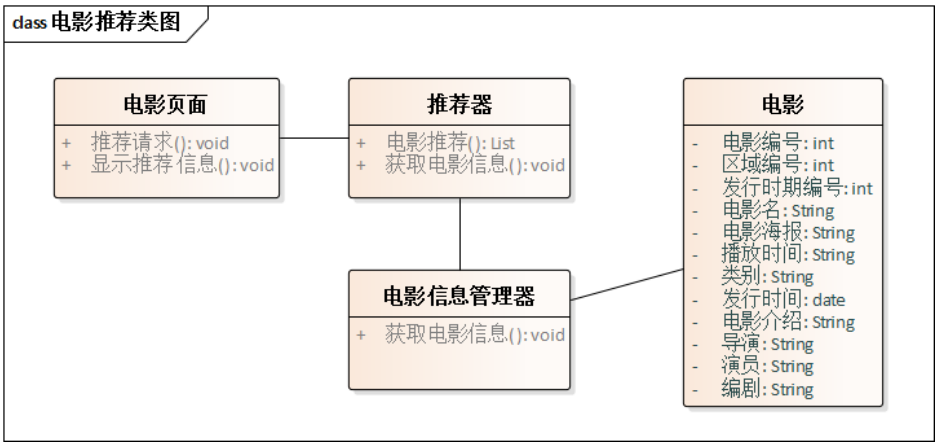

图 4-14 “电影推荐”用例的类图

(6) 电影评论（A011）

表 4-8 展示了“电影评论”用例所包含的接口类、控制类与实体类。其中接口类包括电影详情页面，控制类包括电影信息管理器和评论信息管理器，实体类包括电影类和电影评论类。

表 4-8 “电影评论”用例的接口类、控制类、实体类

接口类	控制类	实体类
电影详情页面	评论信息管理器	电影评论
	电影信息管理器	电影

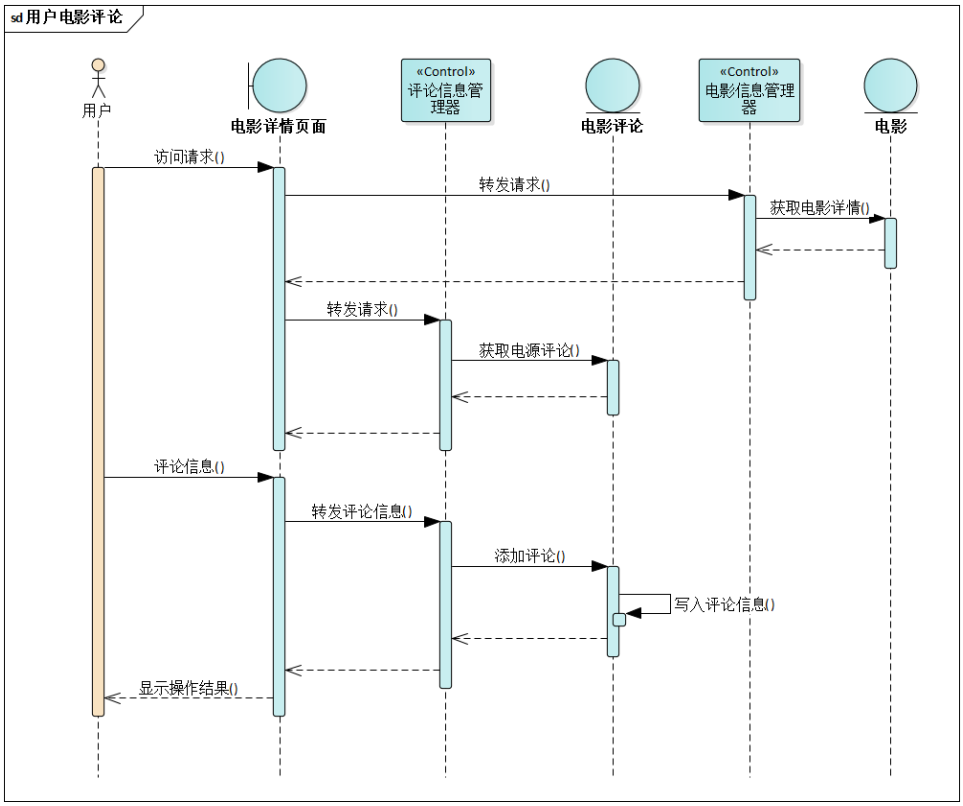

图 4-15 “电影评论”用例的顺序图

图 4-15 描述了用户电影评论的前端操作逻辑和后台处理逻辑。“电影评论”用例类与类的关系图，如图 4-16 所示。

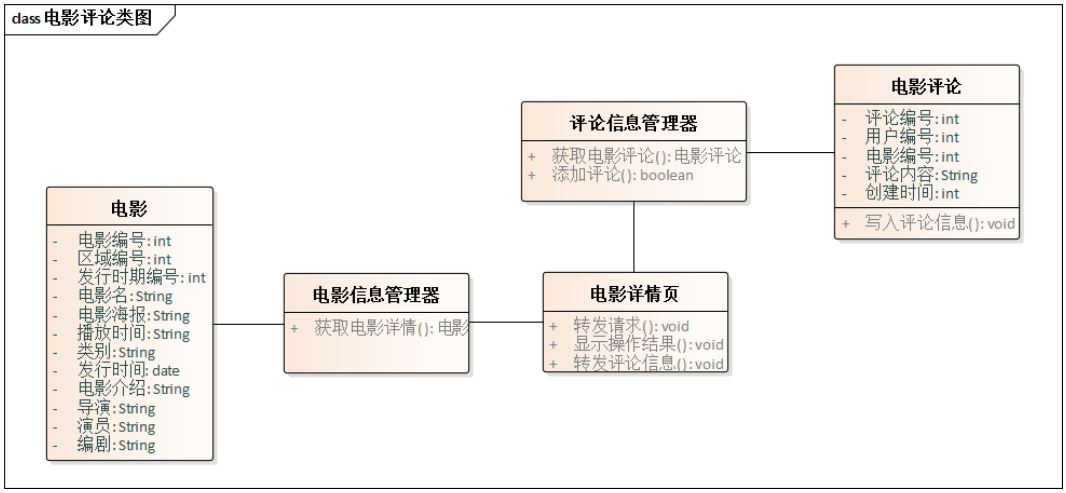

图 4-16 “电影评论”用例的类图

(7) 管理电影评论（B001）

表 4-9 展示了“管理电影评论”用例所包含的接口类、控制类与实体类。其中接口类包括评论管理页面，控制类包括电影信息管理器、评论信息管理器，实体类包括电影类、电影评论类。

表 4-9 “管理电影评论”用例的接口类、控制类、实体类

接口类	控制类	实体类
评论管理页面	评论信息管理器	电影评论
	电影信息管理器	电影

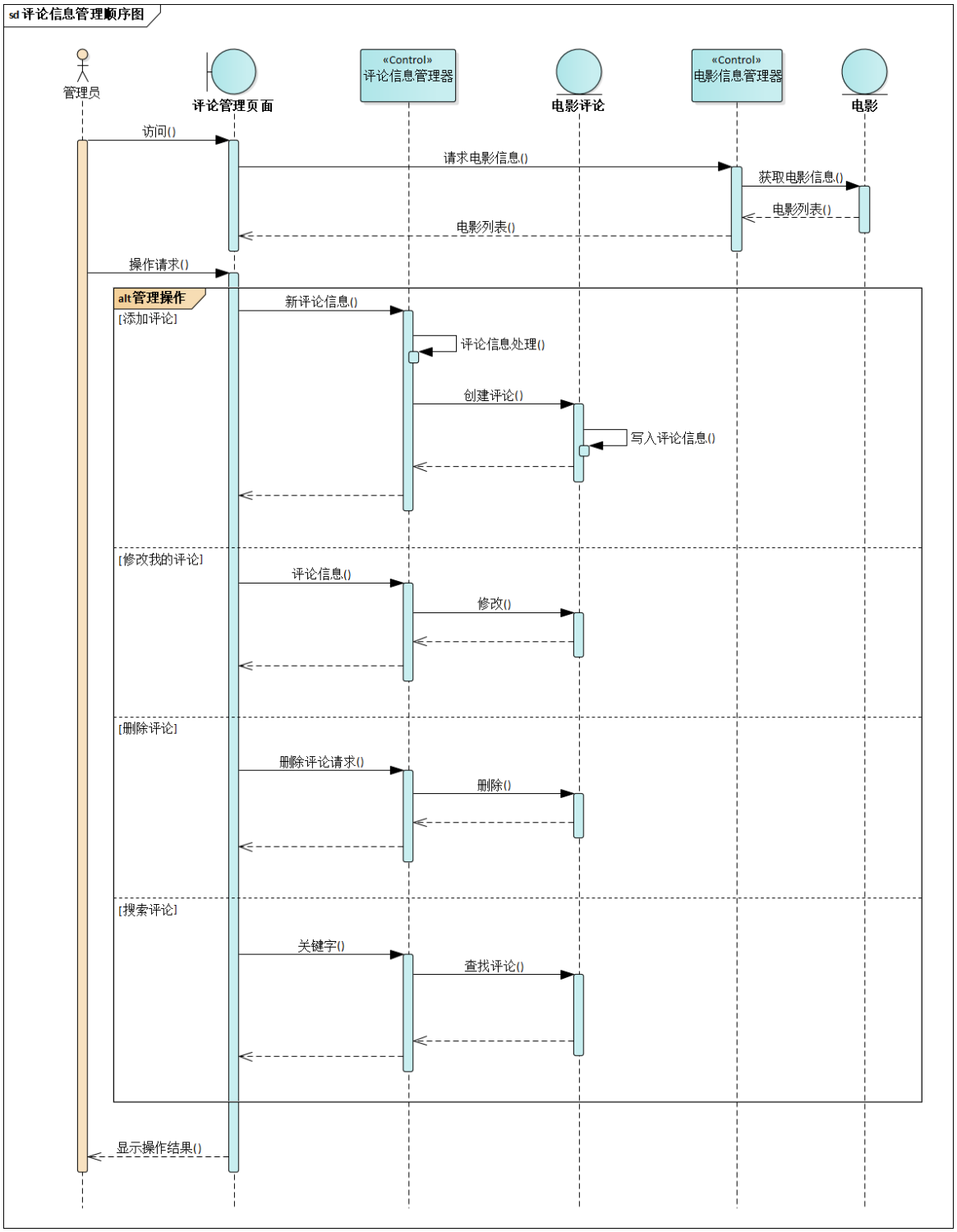

图 4-17　“管理电影评论”用例的顺序图

图 4-17 描述了管理员管理电影评论的前端操作逻辑和后台处理逻辑。“管理电影评论”用例的类图，如图 4-18 所示。

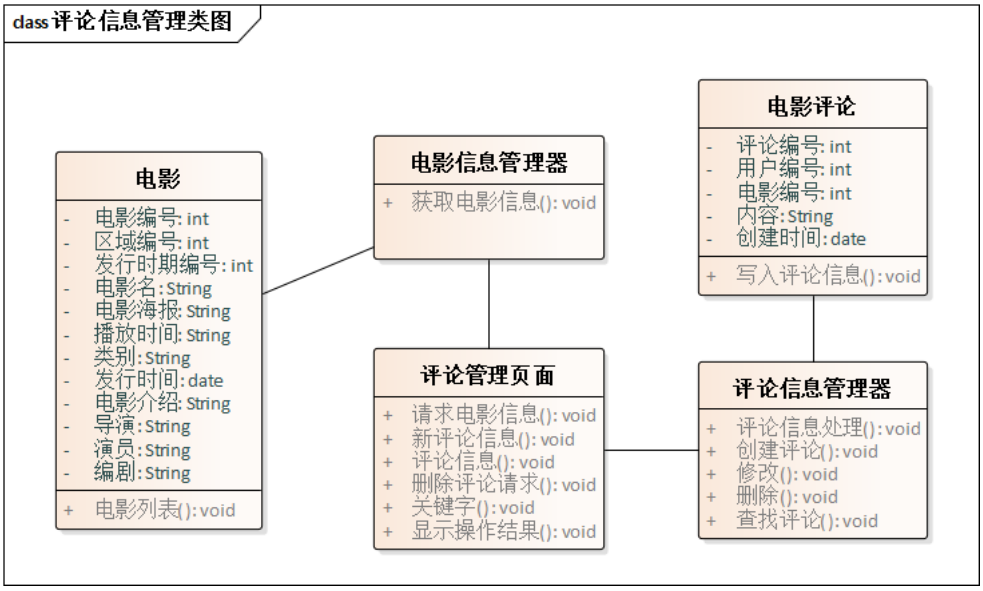

图 4-18 “管理电影评论”用例的类图

(8) 管理用户公告（B002）

表 4-10 展示了“管理用户公告”用例所包含的接口类、控制类与实体类。其中接口类包括公告管理页面，控制类包括公告信息管理器、用户信息管理器、用户通知明细控制器，实体类包括用户通知类、用户类、通知明细类。

表 4-10 “管理用户公告”用例的接口类、控制类、实体类

接口类	控制类	实体类
公告管理页面	公告信息管理器	用户通知
	用户信息管理器	用户
	用户通知明细控制器	通知明细

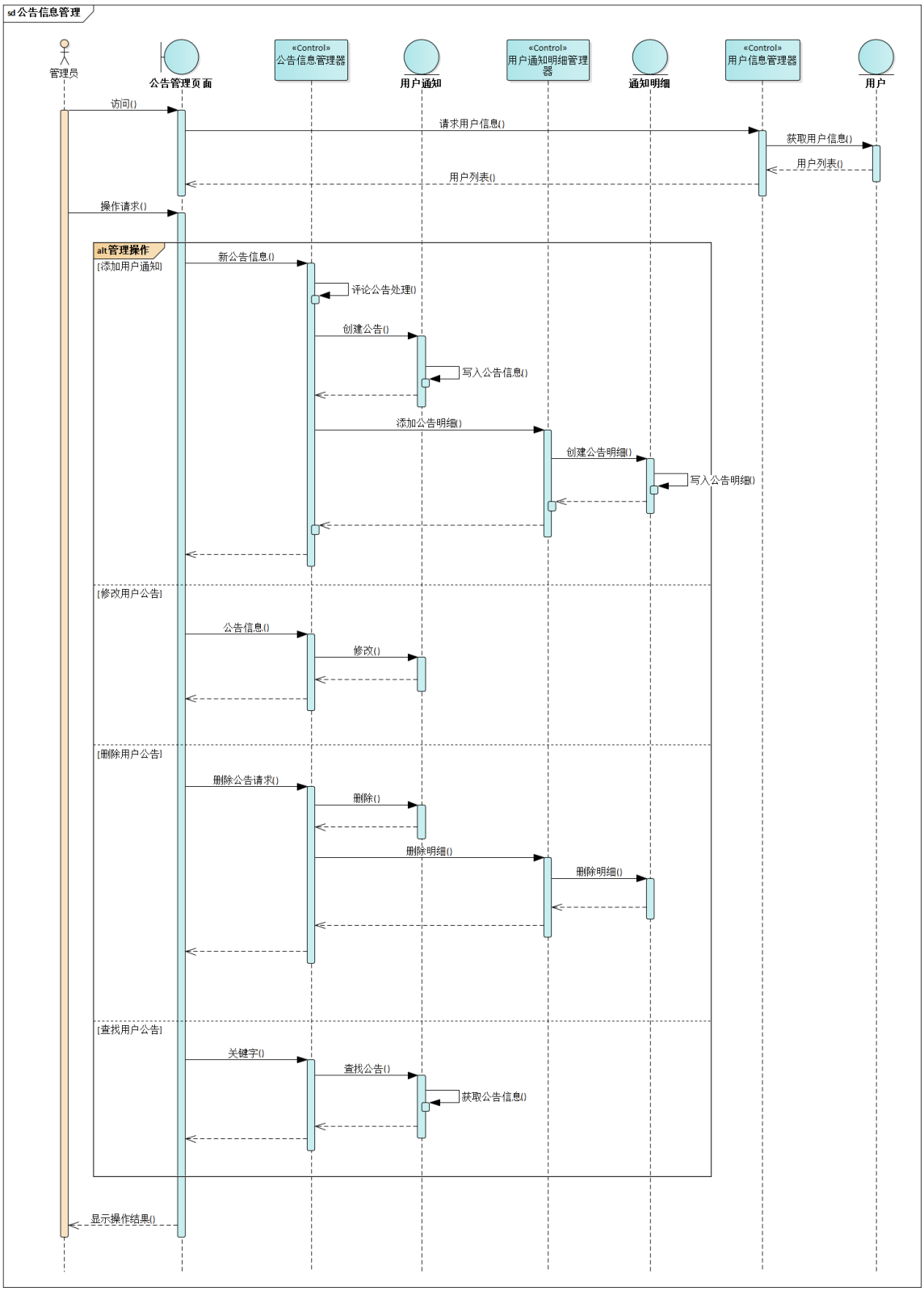

图 4-19　“管理用户公告”用例的顺序图

图 4-19 描述了管理员管理用户公告的前端操作逻辑和后台处理逻辑。图 4-20 则展示了“管理用户公告”用例的类图。

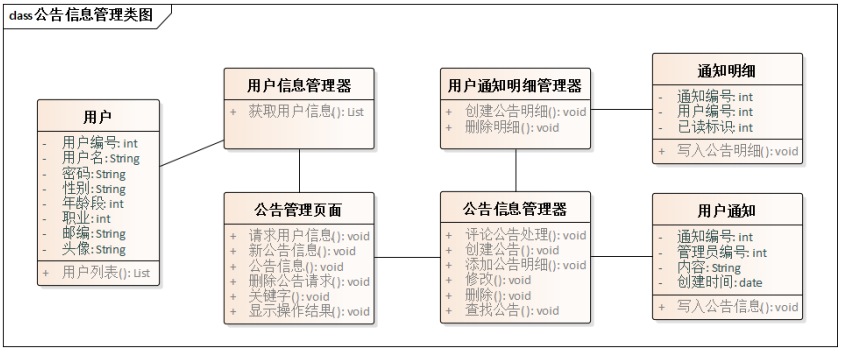

图 4-20　“管理用户公告”用例的类图

(9) 维护用户信息（B003）

表 4-11 展示了“维护用户信息”用例所包含的接口类、控制类与实体类。其中接口类包括用户管理页面，控制类包括用户信息管理器，实体类包括用户类。

表 4-11　“维护用户信息”用例的接口类、控制类、实体类

接口类	控制类	实体类
用户管理页面	用户信息管理器	用户

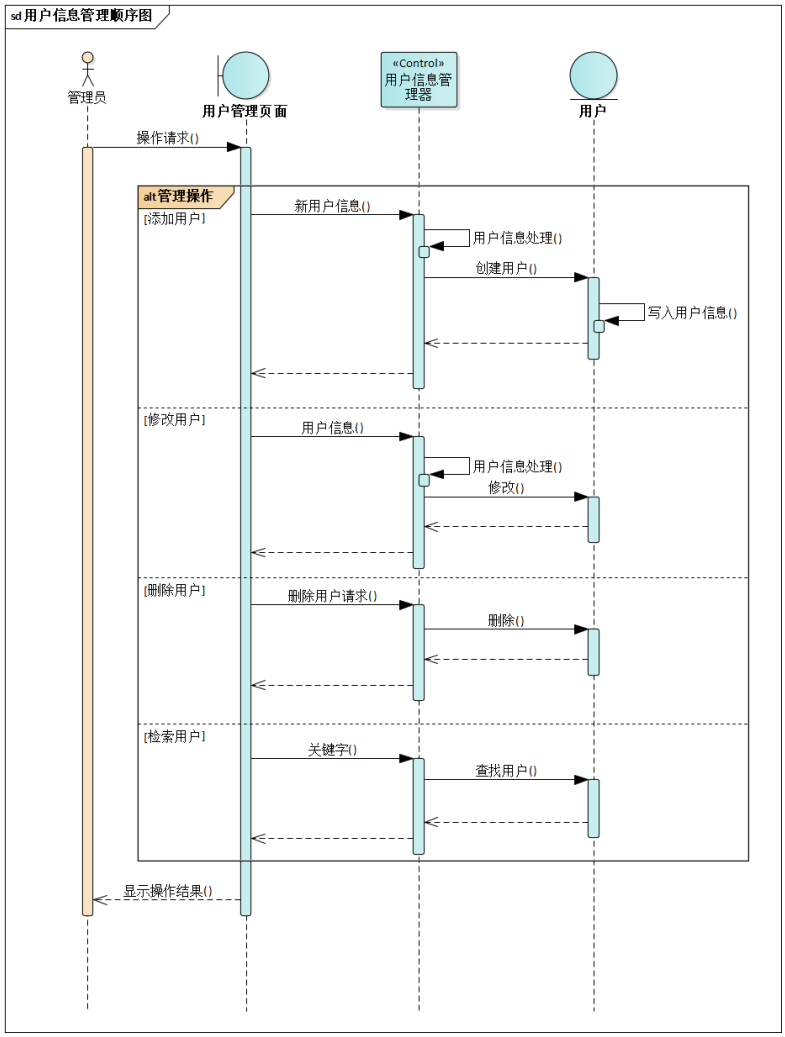

图 4-21　“维护用户信息”用例的顺序图

图 4-21 描述了管理员维护用户信息的前端操作逻辑和后台处理逻辑。图 4-22 则展示了“维护用户信息”用例的类图。

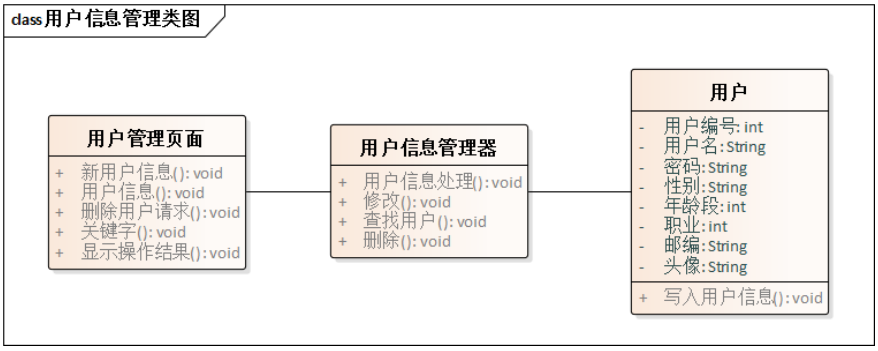

图 4-22 “维护用户信息”用例的类图

(10) 维护电影信息（B004）

表 4-12 展示了“维护电影信息”用例所包含的接口类、控制类与实体类。其中接口类包括电影管理页面，控制类包括电影信息管理器、类别明细管理器，实体类包括电影类、电影类别明细类。

表 4-12 “维护电影信息”用例的接口类、控制类、实体类

接口类	**控制类**	**实体类**
电影管理页面	电影信息管理器	电影
	类别明细管理器	电影类别明细

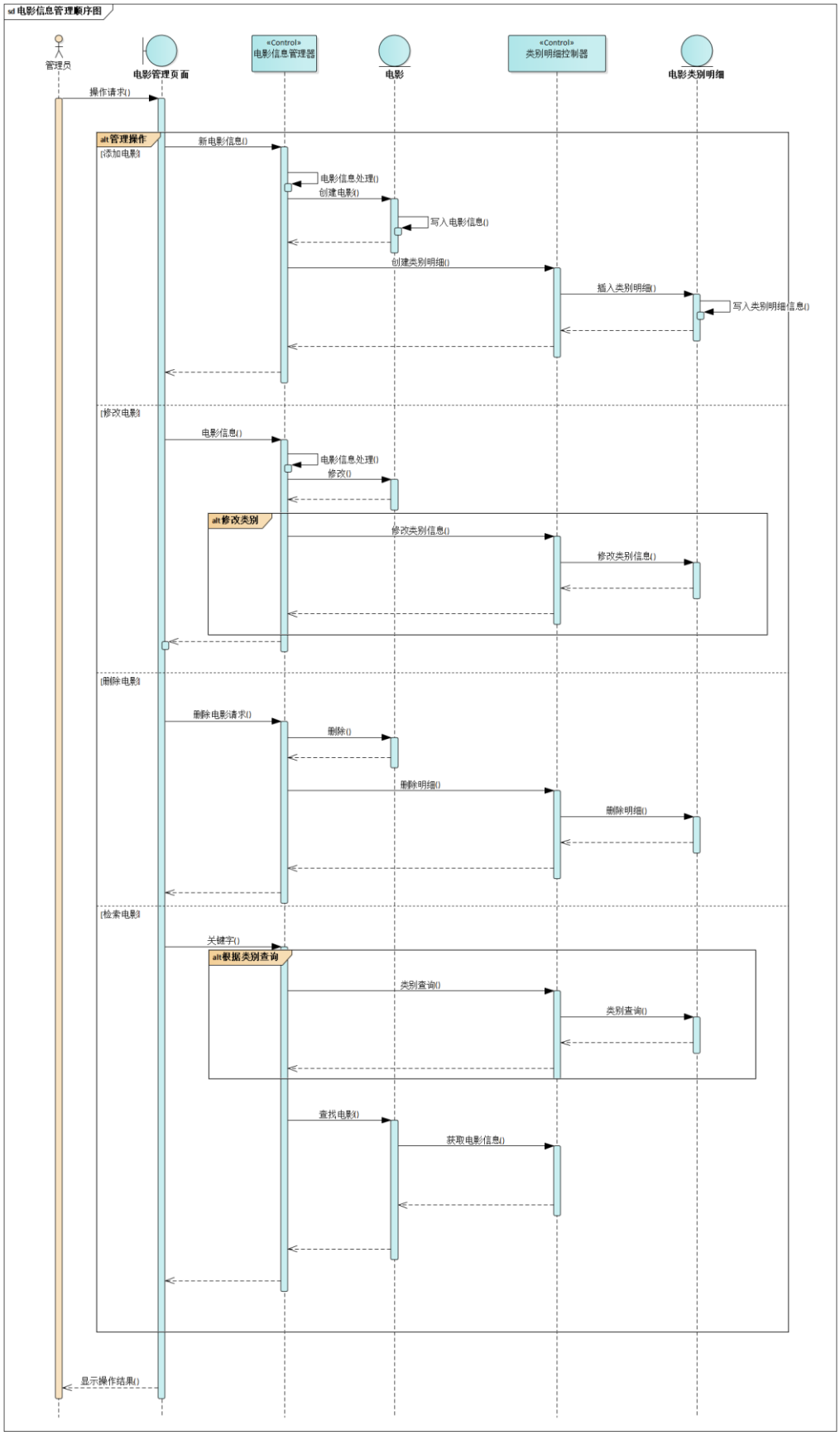

图 4-23　“维护电影信息”用例的顺序图

图 4-23 描述了管理员维护电影信息的前端操作逻辑和后台处理逻辑。“维护电影信息”用例中类与类的关系图，如图 4-24 所示。

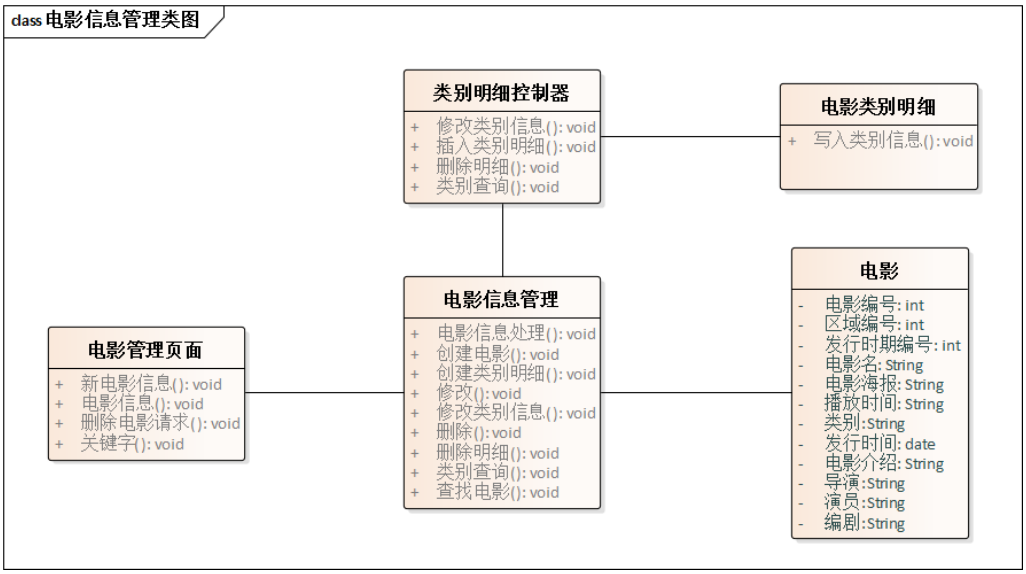

图 4-24 “维护电影信息”用例的类图

(11) 权限管理（B009）

表 4-13 展示了“权限管理”用例所包含的接口类、控制类与实体类。其中接口类包括权限管理页面，控制类包括权限管理器、管理员信息管理器、权限明细管理器，实体类包括权限类、管理员类和权限明细类。

表 4-13 “权限管理”用例的接口类、控制类、实体类

接口类	**控制类**	**实体类**
权限管理页面	权限管理器	权限
	管理员信息管理器	管理员
	权限明细管理器	权限明细

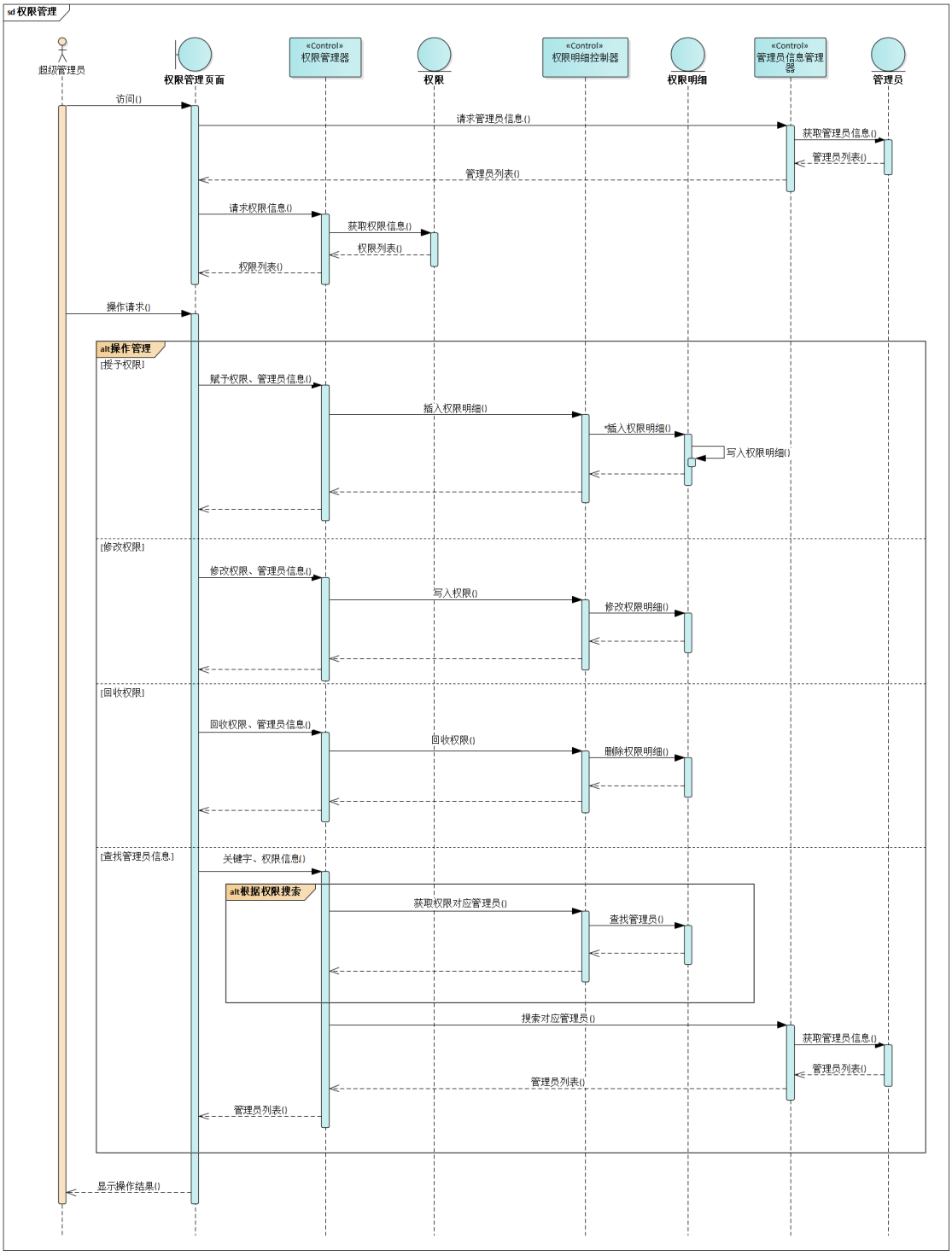

图 4-25　“权限管理”用例的顺序图

图 4-25 描述了超级管理员维护后台管理端权限信息的前端操作逻辑和后台处理逻辑。“权限管理”用例中类与类的关系图，如图 4-26 所示。

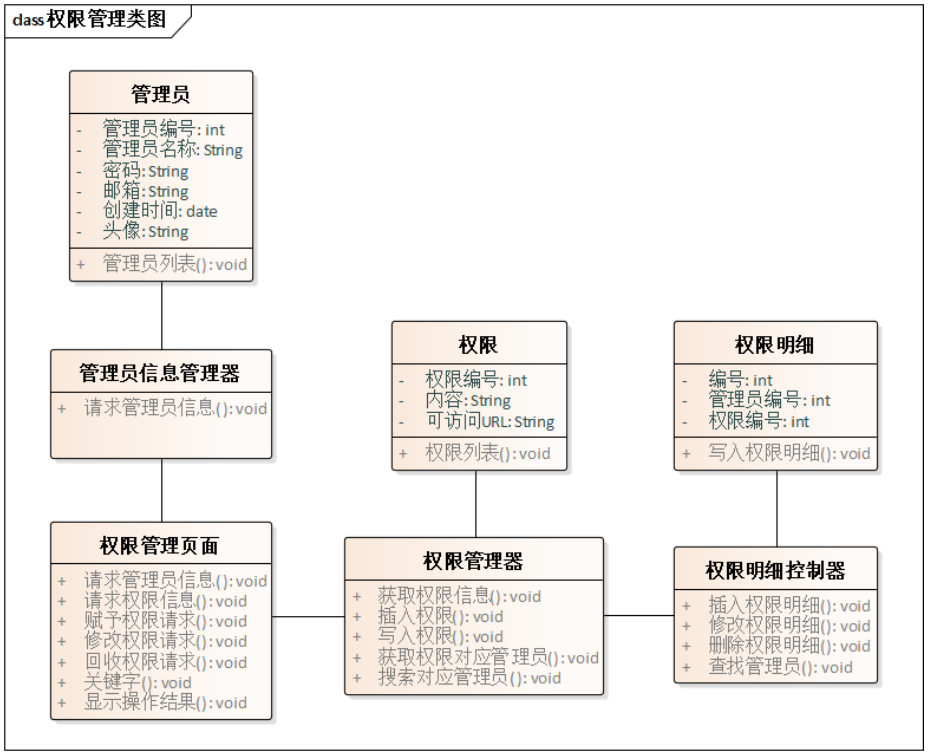

图 4-26　“权限管理”用例的类图

4.2.3 算法层设计

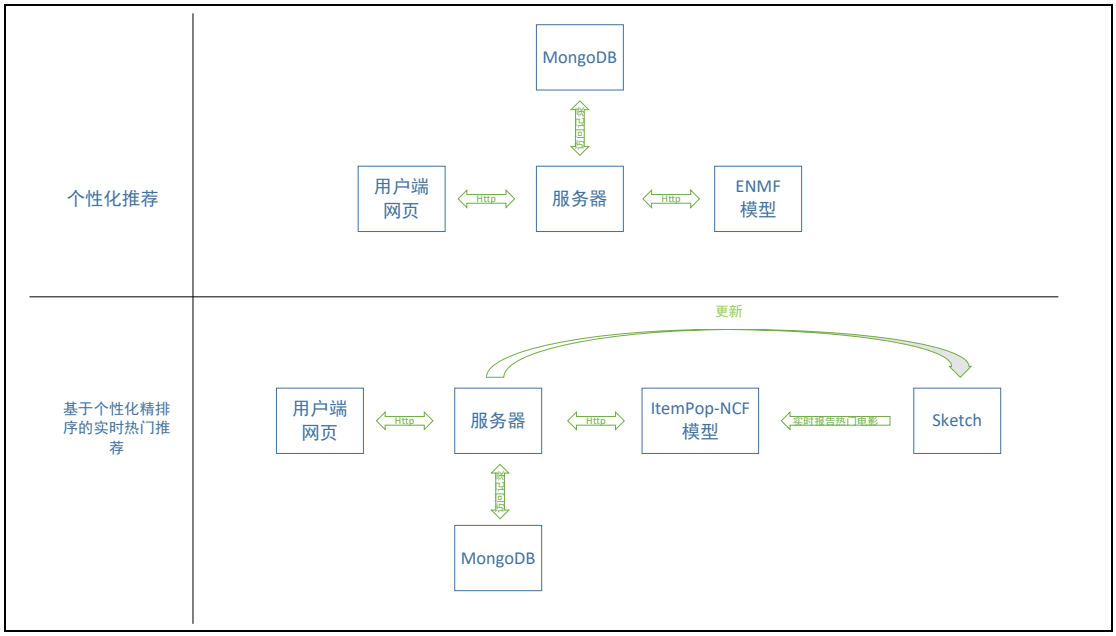

图 4-27　算法层设计图

图 4-27 展示了算法层的设计。主要分为个性化分析和基于个性化精排序的实时热门推荐两个模块。

对于个性化推荐部分，本文选取在实验中效果最好的 ENMF 模型。在此，数据集划分没有和实验部分一样划分一个验证集，即每个用户的正例集只随机取一个交互分别作为测试集。调完参之后，整个数据集再用最优超参训模型，得到最终推荐模型。最终模型部署在 Flask 上，服务端通过 Http 请求调用，以用户 ID 作为 ENMF 模型 user 端输入，返回 json 格式的推荐数据。

对于基于个性化精排序的实时热门推荐，本文选取实验中效果最好的 ItemPop-NCF 模型。模型的训练和部署与上述个性化推荐部分一致。不同的是，在 Flask 上额外部署了用户流 Sketch，并能根据用

户访问流动态更新，返回实时的热门电影。以用户 ID 作为 NCF 模型 user 端输入，热门电影 ID 作为 NCF 模型 item 端特定输入，返回更高质量的热门推荐。

对于网站的访问记录流，系统部署了 MongoDB 进行存取，用于用户历史记录的查询、日后的用户行为分析及模型的辅助训练，为系统增加了一定的扩展性。

4.2.4 数据库设计

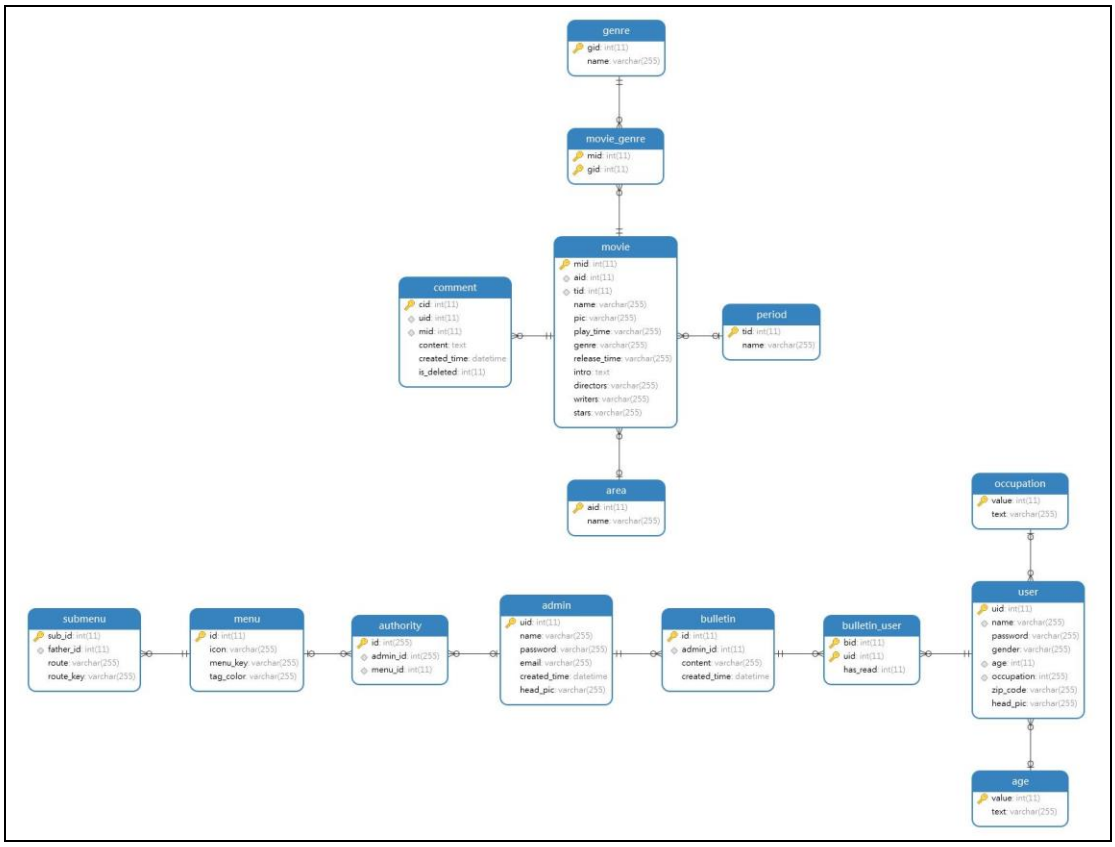

图 4-28　MySQL 数据库 ERD 图

图 4-28 展示了本电影推荐系统的主要业务功能的实体关系。基于实体关系，数据库的表设计如下：

（1）user 表

表 4-14 为用户表，主键为 uid 字段，age、occupation 字段作为外键，分别与 age 表 value 字段、occupation 表 value 字段关联。该表主要为登录、注册、用户信息维护等功能提供用户数据支持。表约有 1 万条用户数据。

表 4-14　user 表

字段名	数据类型	说明	约束
uid	int	用户编号	主键
name	varchar	用户名(登录账号）	not null、unique
password	varchar	密码	not null
gender	varchar	性别 M 或 F	
age	int	年龄段编号	外键
occupation	int	职业编号	外键
zip_code	varchar	邮政编码	
head_pic	varchar	头像 url	

（2）age 表

表 4-15 为年龄段表，用于存储、维护年龄段信息，其中 value 字段为表的主键，一个用户至多对应一个年龄段，一个年龄段可以被多个用户对应。该表主要为登录、注册、用户信息维护等功能提供年龄

段数据支持。

表 4-15　age 表

字段名	数据类型	说明	约束
value	int	年龄段编号	主键
text	varchar	年龄段描述	not null

（3）occupation 表

表 4-16 为职业表，用于存储、维护职业信息，其中 value 字段为表的主键，一个用户至多对应一种职业，一种职业可以被多个用户对应。该表主要为登录、注册、用户信息维护等功能提供职业数据支持。

表 4-16　occupation 表

字段名	数据类型	说明	约束
value	int	职业编号	主键
text	varchar	职业描述	not null

（4）movie 表

表 4-17 为电影表，用于存储、维护电影信息，其中 mid 字段为表的主键，aid、tid 字段作为外键，分别与 area 表 aid 字段、genre 表 gid 字段关联。表内的 genre 字段是提高类别查询效率而设立的额外字段。该表主要为电影信息维护、电影推荐等功能提供电影数据支持，表约有 20 万条电影数据。

表 4-17　movie 表

字段名	数据类型	说明	约束
mid	int	电影编号	主键
aid	int	区域编号	外键
tid	int	发行时期编号	外键
name	varchar	电影名	
pic	varchar	电影海报 url	
play_time	varchar	播放时长	
genre	varchar	类别（便于查询）	
release_time	varchar	发行时间	
intro	text	电影简介	
directors	varchar	导演	
writers	varchar	编剧	
stars	varchar	演员	

（5）area 表

表 4-18 为电影地区表，用于存储、维护电影地区信息，其中 aid 字段为表的主键，一部电影至多对应一个类别，一个类别可以被多部电影对应。该表主要为电影信息维护、电影分类目录检索等功能提供电影地区数据支持。

表 4-18　area 表

字段名	数据类型	说明	约束
aid	int	地区编号	主键
name	varchar	地区名	not null

（6）genre 表

表 4-19 为电影类别表，用于存储、维护电影类别信息，其中 gid 字段为表的主键，一部电影可以

对应多个类别，一个类别可以被多部电影对应。该表主要为电影信息维护、电影分类目录检索等功能提供电影类别数据支持。

表 4-19　genre 表

字段名	数据类型	说明	约束
gid	int	类别编号	主键
name	varchar	类别名	not null

（7）movie_genre 表

表 4-20 为电影类别明细表，用于存储、维护电影类别明细，其中 mid、gid 字段为表的混合主键，且作为外键，分别与 movie 表 mid 字段、genre 表 gid 字段关联。

表 4-20　movie_genre 表

字段名	数据类型	说明	约束
mid	int	电影编号	主键、外键
gid	int	类别编号	主键、外键

（8）period 表

表 4-21 为电影发行时期表，用于存储、维护电影发行时期信息，其中 tid 字段为表的主键，一部电影至多对应一个发行时期，一个发行时期可以被多部电影对应。该表主要为电影信息维护、电影分类目录检索等功能提供电影发行时期数据支持。

表 4-21　period 表

字段名	数据类型	说明	约束
tid	int	发行时期编号	主键
name	varchar	发行时期	not null

（9）comment 表

表 4-22 为电影评论表，用于存储、维护电影评论信息，其中 cid 字段为表的主键，mid 字段作为外键，与 movie 表 mid 字段对应，uid 字段为评论的发出者的编号，可以为用户编号或管理员编号。一部电影可对应多个评论，一个评论至多被一部电影对应。该表主要为评论信息维护、用户评论等功能提供电影评论数据支持。

表 4-22　comment 表

字段名	数据类型	说明	约束
cid	int	评论编号	主键
mid	int	电影编号	外键
uid	int	评论者编号(用户编号或管理员编号)	
content	text	评论内容	
created_time	datetime	创建时间	
is_deleted	int	删除表示(0-未删除 1-已删除)	

（10）admin 表

表 4-23 为管理员表，用于存储、维护管理员信息，其中 uid 字段为表的主键。该表主要用于管理员信息维护、权限管理、管理员登录等功能。

表 4-23　admin 表

字段名	数据类型	说明	约束
uid	int	管理员编号	主键
name	varchar	用户名(登录账号)	not null、unique

password	varchar	密码	not null
email	varchar	邮箱	
created_time	datetime	创建时间	
head_pic	varchar	头像 url	

（11）menu 表

表 4-24 为管理端父级菜单表，用于存储、维护管理端父级菜单信息，其中 id 字段为表的主键。表的内容与管理端的网页显示相关。

表 4-24 menu 表

字段名	数据类型	说明	约束
id	int	父级菜单编号	主键
icon	varchar	图标	
menu_key	varchar	父级菜单标识键	not null
tag_color	varchar	颜色	

（12）submenu 表

表 4-25 为管理端子级菜单表，用于存储、维护管理端子级菜单信息，其中 sub_id 字段为表的主键，father_id 字段作为外键与 menu 表的 id 字段关联。表的内容与管理端的网页显示相关，并存储了菜单对应功能的路由信息。

表 4-25 submenu 表

字段名	数据类型	说明	约束
sub_id	int	子级菜单编号	主键
father_id	int	父级菜单编号	外键
route	varchar	路由 url	
route_key	varchar	子级菜单标识键	not null

（13）authority 表

表 4-26 为权限明细表，主键为 id 字段，admin_id、menu_id 字段作为外键分别与 admin 表的 uid 字段、menu 表的 id 字段关联。一个管理员可以对应多个菜单功能的操作权限，一个菜单功能的操作权限可以被多个管理员对应。权限的动态管理功能由 admin 表、menu 表、submenu 表、authority 表共同实现。

表 4-26 authority 表

字段名	数据类型	说明	约束
id	int	权限编号	主键
admin_id	int	管理员编号	外键
menu_id	int	父级菜单编号	外键

（14）bulletin 表

表 4-27 为用户通知公告表，主键为 id 字段，admin_id 字段作为外键与 admin 表的 uid 字段关联。一名管理员能发布多个用户的通知或公告，一个通知或公告至多被一名管理员对应。该表主要用于存储、提供、维护管理员发送的用户通知公告信息。

表 4-27 bulletin 表

字段名	数据类型	说明	约束
id	int	通知公告编号	主键
admin_id	int	发布者编号(管理员)	外键

content	varchar	内容	not null
created_time	datetime	创建时间	

（15）bulletin_user 表

表 4-28 为用户通知公告明细表，用于存储、维护用户通知公告明细信息，其中 bid、uid 字段为表的混合主键，同时作为外键分别与 bulletin 表 bid 字段、user 表的 uid 字段关联。一个由管理员发布的通知公告可以与多个用户对应，一个用户可以对应多个通知公告。已读标识默认为 0，当通知公告被用户阅读后，设置为 1。

表 4-28　bulletin 表

字段名	数据类型	说明	约束
bid	int	通知公告编号	主键、外键
uid	int	用户编号	主键、外键
has_read	int	阅读标识(0-未阅读 1-已阅读)	默认 0

4.3 系统实现

4.3.1 用户端实现

（1）登录/注册页

登录页如图 4-29 所示，勾选 Admin Access 输入管理员账号密码后可登录到后台管理端。非勾选时为普通用户登录，跳转到用户端首页。点击 Register now!连接后，会从网站右侧弹出注册表单（如图 4-29）。键入相应信息后，即可完成注册。用注册好的账号密码可在非勾选 Admin Access 的状态登录到用户端。

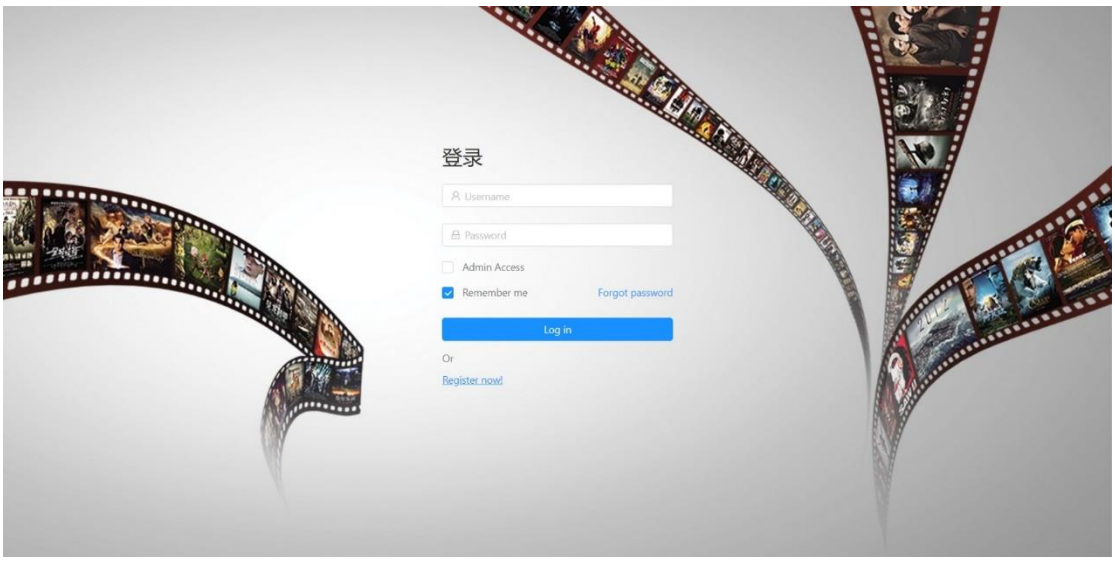

图 4-29 登录页

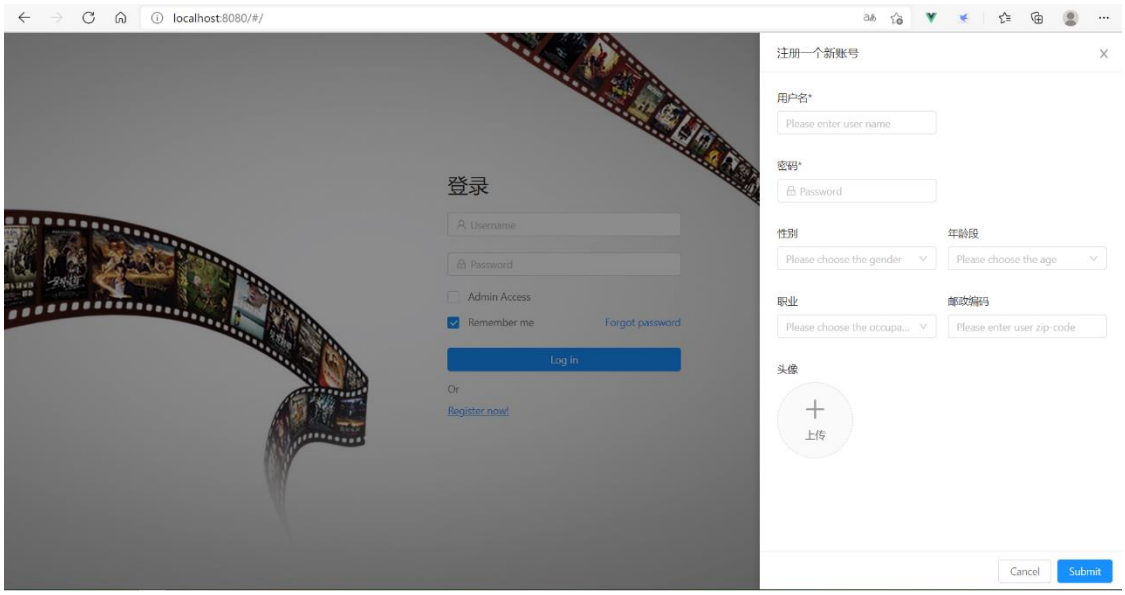

图 4-30 注册页

(2) 网站首页

首页如图 4-31 所示，主要包含轮播图、猜你喜欢、热门推荐功能。操作顶部导航栏可以触发相应的功能，如进入影视库、关键字搜索、用户信息修改、通知查看（顶部导航栏的数字气泡表示未读的通知数目）、历史记录查看、登出等。点击推荐电影的海报，可以进入相应电影的详细页面。

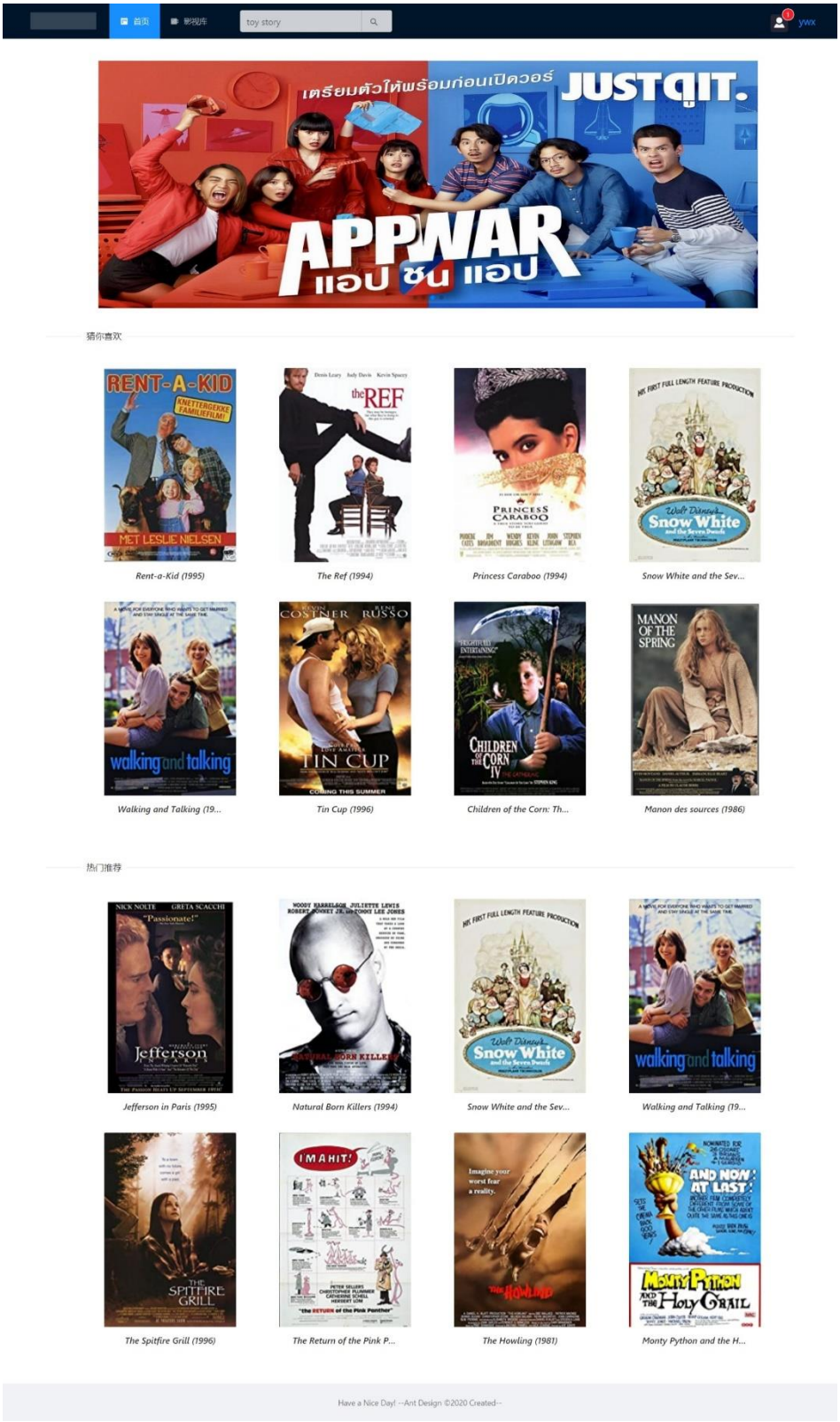

图 4-31 网站首页-猜你喜欢-热门推荐

(3) 分类目录页

分类目录页如图 4-32 所示，主要为电影库的分类目录检索（地区、时间、类别的组合）。操作顶部导航栏仍然可以触发相应的功能。点击电影的海报，可以进入相应电影的详细页面。

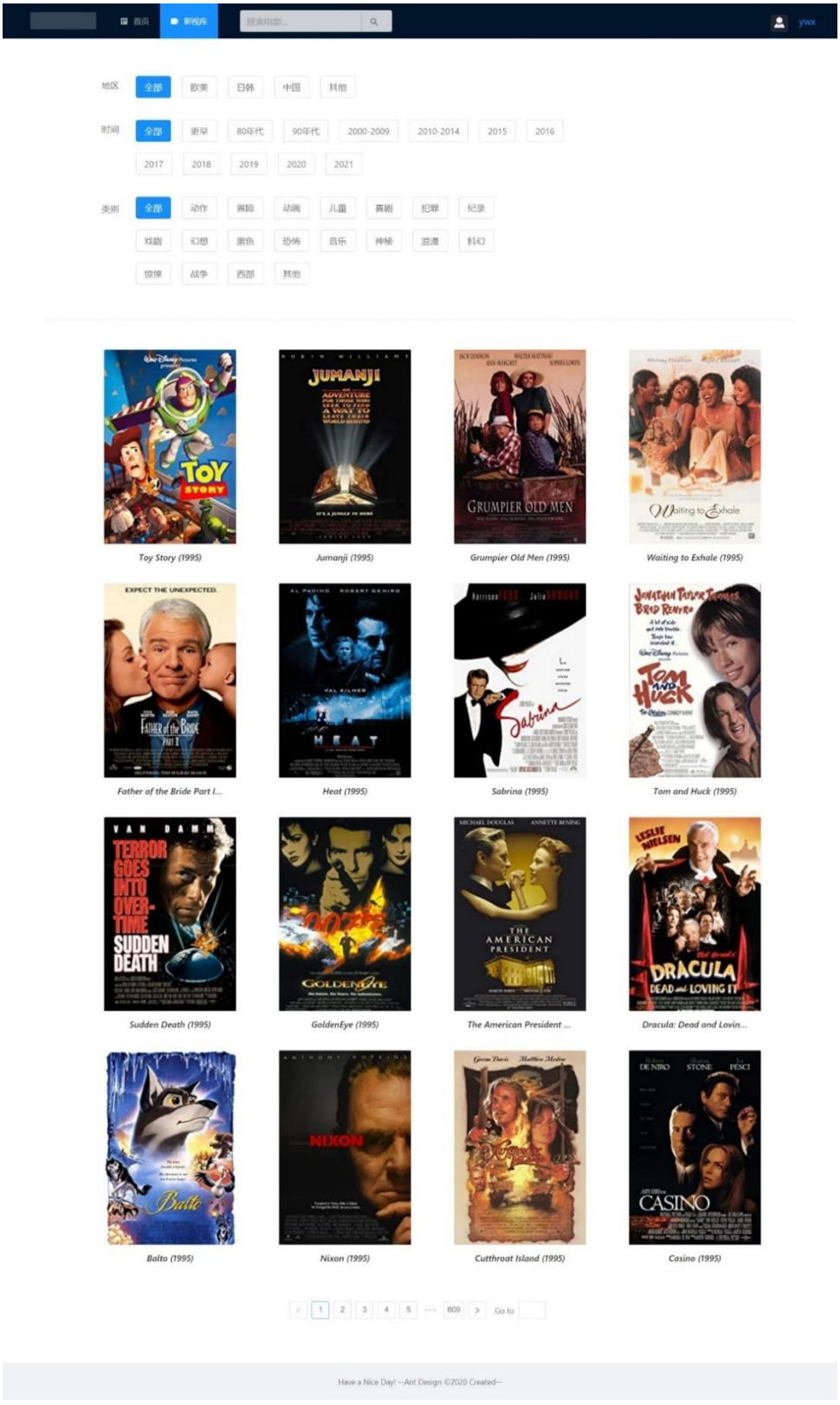

图 4-32 影视库-分类目录

(4) 电影详细页与用户评论

电影详细页如图 4-33 所示，主要包含电影详细信息展示、电影评论功能。

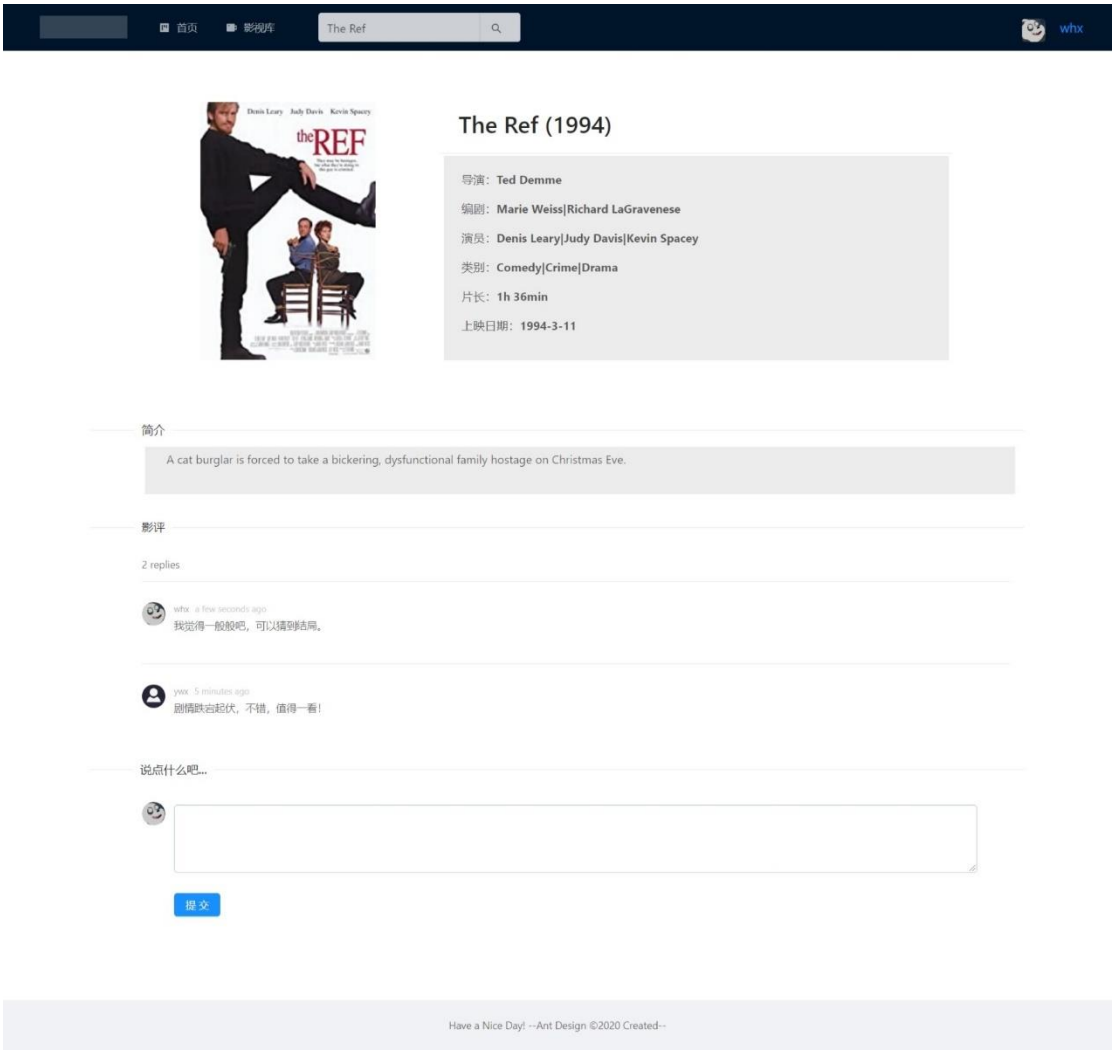

图 4-33 电影详情页-用户电影评论

(5) 关键字检索

关键字检索功能如图 4-34 所示。输入电影名/类别/简介/导演/演员的关键字，即可模糊检索出相应电影，并在右侧显示。点击右侧电影信息，即可跳转到相应电影的详细页面。

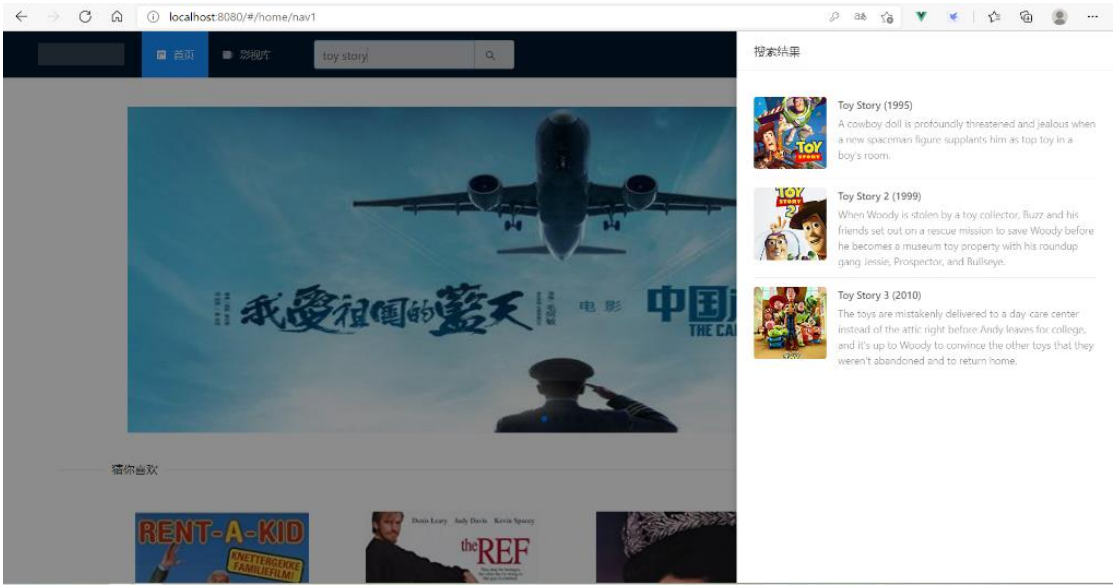

图 4-34 通过关键词检索电影

(6) 系统通知

系统通知功能如图 4-35 所示。点击系统通知，可阅读详情，并标识为已读置底，也可点击全部已读消除顶部导航栏气泡。

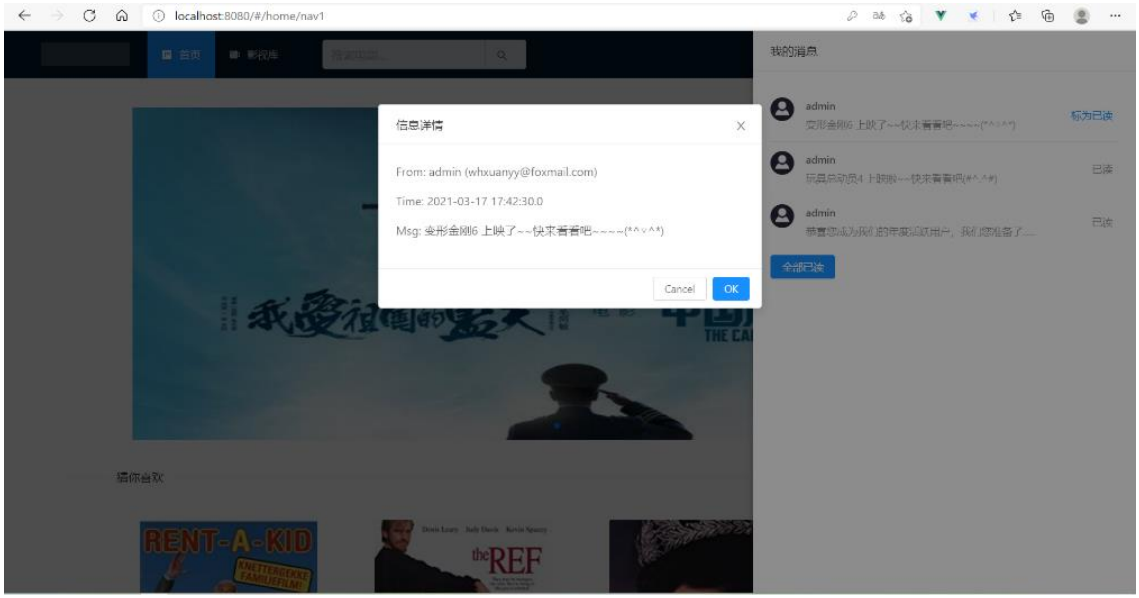

图 4-35 用户通知公告

(7) 历史记录浏览

历史记录浏览功能如图 4-36 所示。浏览记录按时间倒序。点击右侧电影信息，即可跳转到相应电影的详细页面。点击删除可删除单条记录，全部清空可删除全部记录。

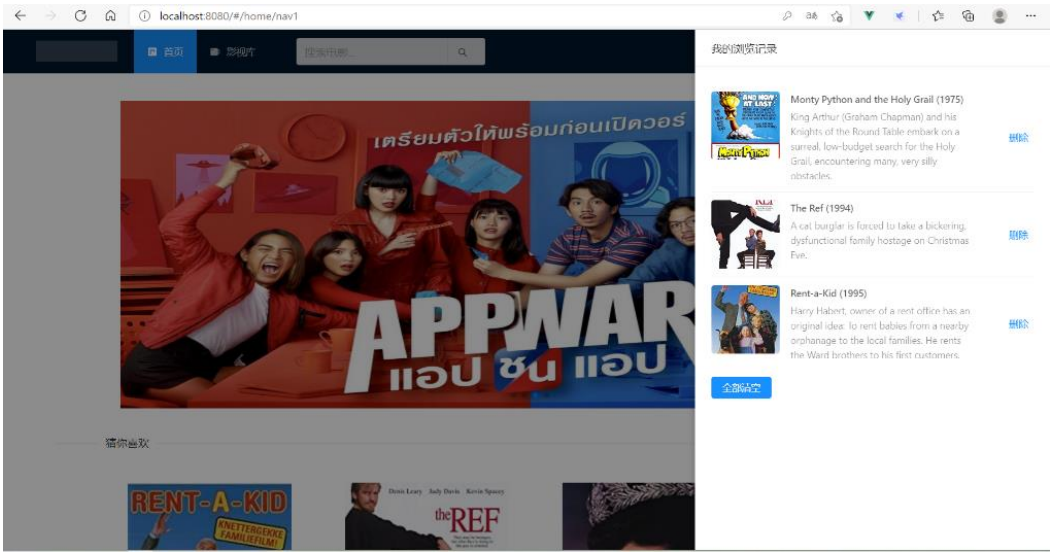

图 4-36 历史浏览记录

4.3.2 后台管理端实现

(1) 用户信息维护

用户信息维护页面如图 4-37 所示。主要包括用户信息的新增、修改、失效、检索操作。查询包括关键字、性别、年龄段、职业的单独及组合查询。

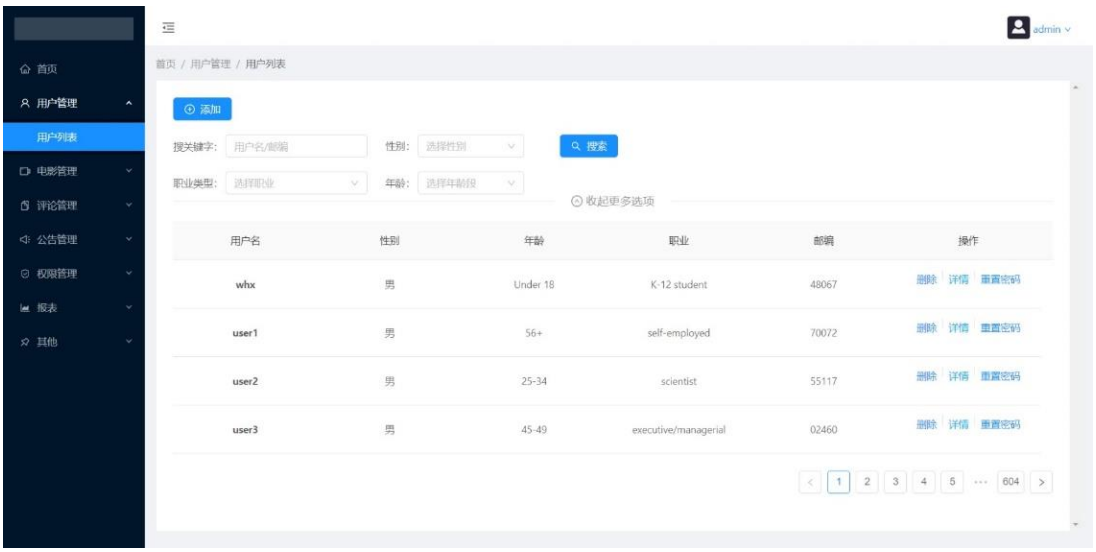

图 4-37 用户信息维护

(2) 电影信息维护

电影信息维护页面如图 4-38 所示。主要包括电影信息的新增、修改、失效、检索操作。查询包括关键字、发行时间区间、电影类别的单独及组合查询。电影详情页如图 4-39 所示，打开修改开关即可进行电影信息修改。

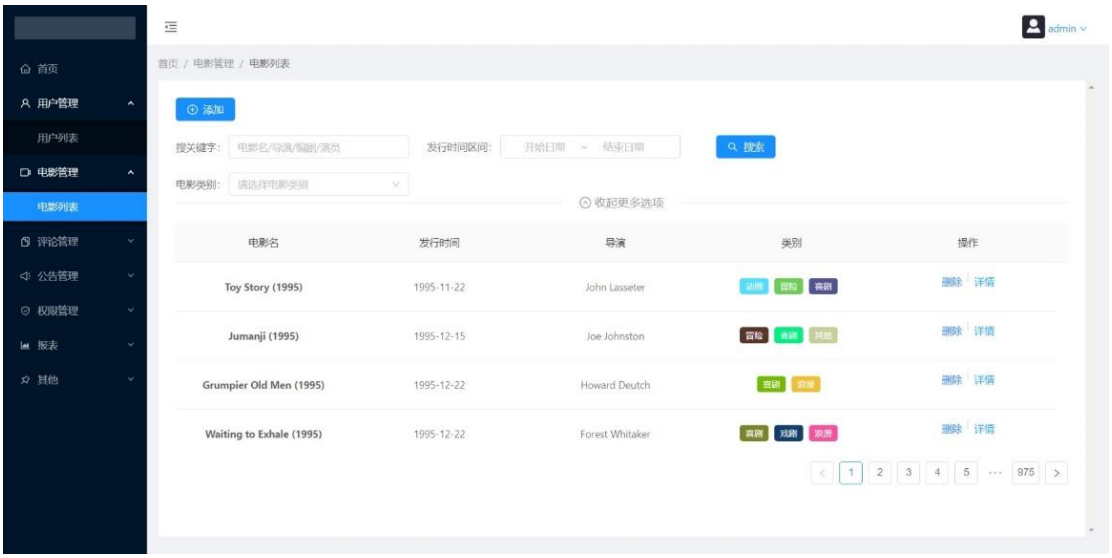

图 4-38 电影信息维护

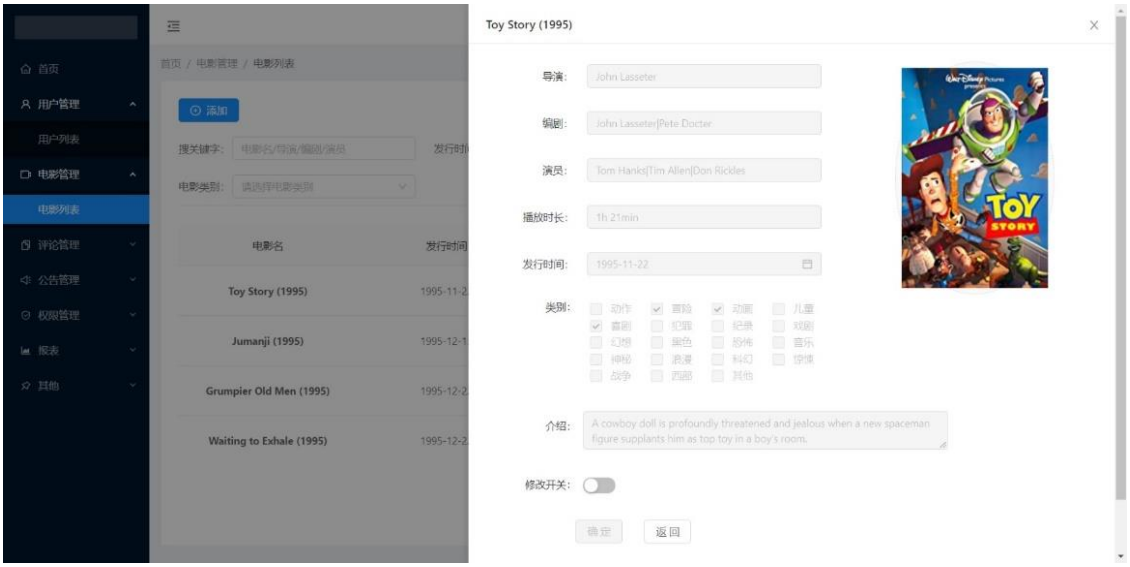

图 4-39 电影信息详情与修改

(3) 评论信息管理

评论信息管理页面如图 4-40 所示。主要用于管理员批量添加、失效、检索评论，及修改管理员批量发布的评论。查询包括关键字、评论时间区间的单独及组合查询。批量评论发布页如图 4-41 所示，页内提供了电影搜索功能，勾选相应电影输入评论内容即可批量评论。

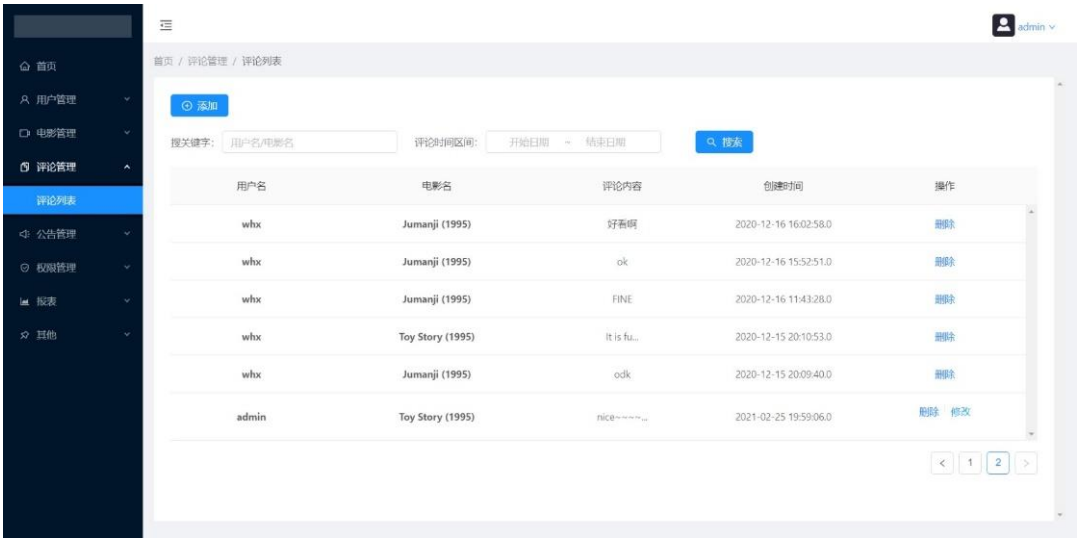

图 4-40 评论信息管理

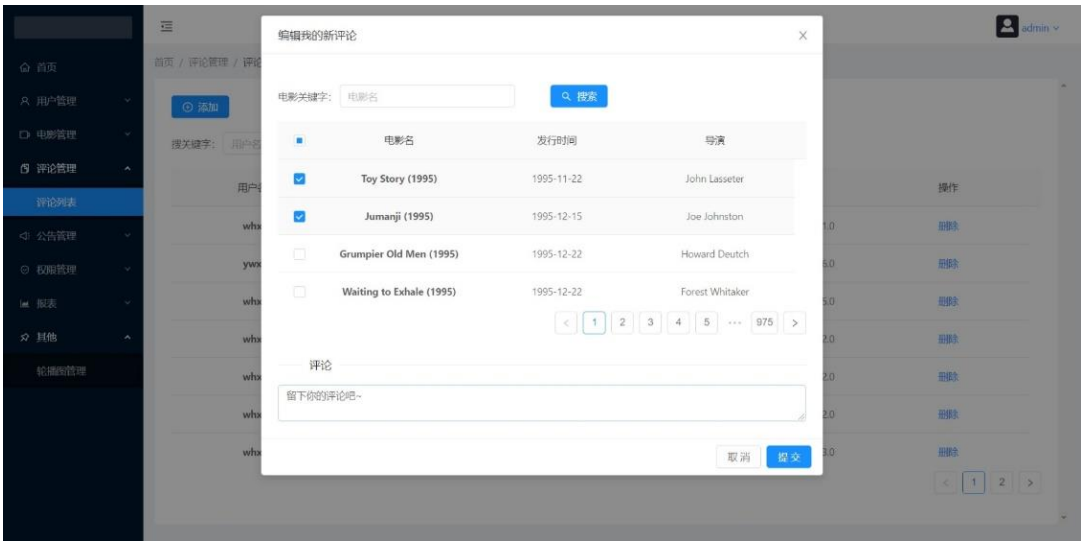

图 4-41 管理员批量发布评论

(4) 用户通知公告管理

用户通知公告页面如图 4-42 所示。主要用于管理员批量添加、失效、修改、检索用户通知公告。批量用户通知公告发布页如图 4-43 所示，页内提供了用户搜索功能，勾选相应用户输入通知公告内容即可批量发布。

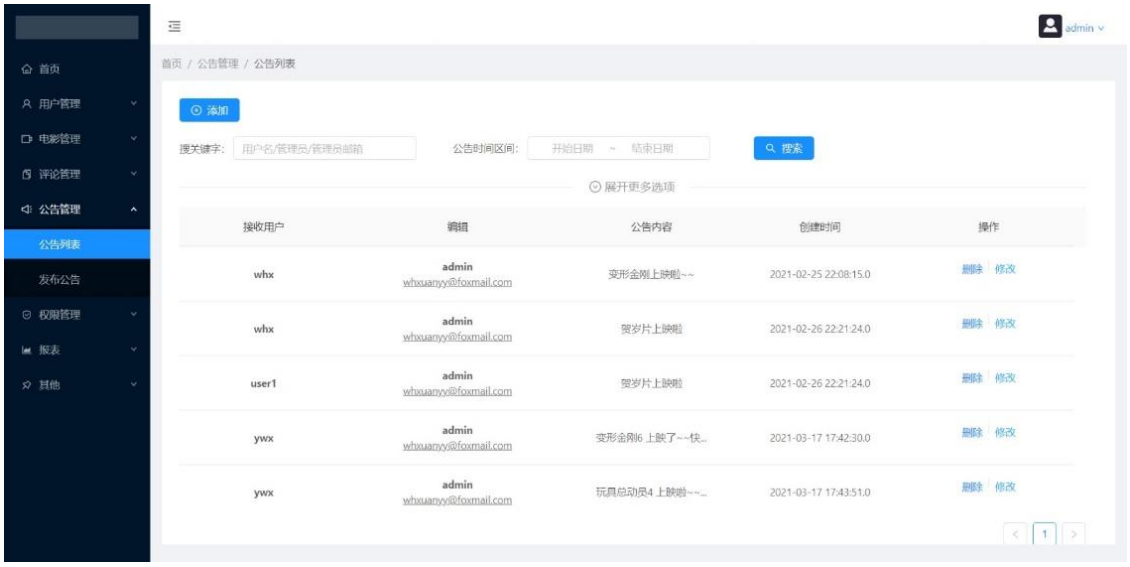

图 4-42 用户通知公告管理

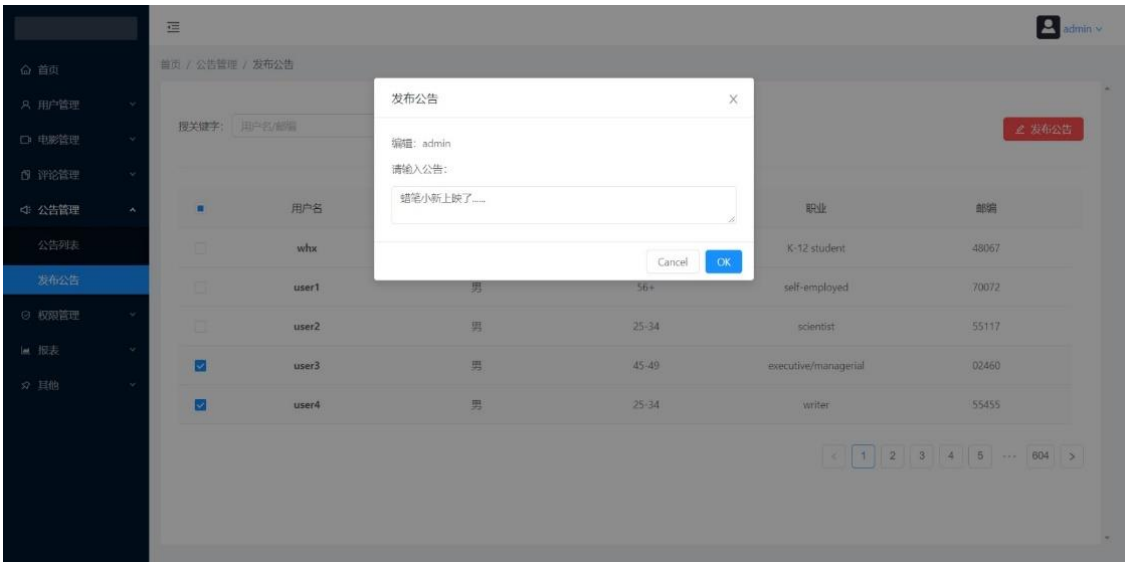

图 4-43 管理员批量发布用户通知公告

(5) 权限管理

权限管理页面如图 4-44~图 4-45 所示，主要包含权限管理、管理员信息管理功能。打开修改开关后，勾选相应权限即可获得相应系统操作权。

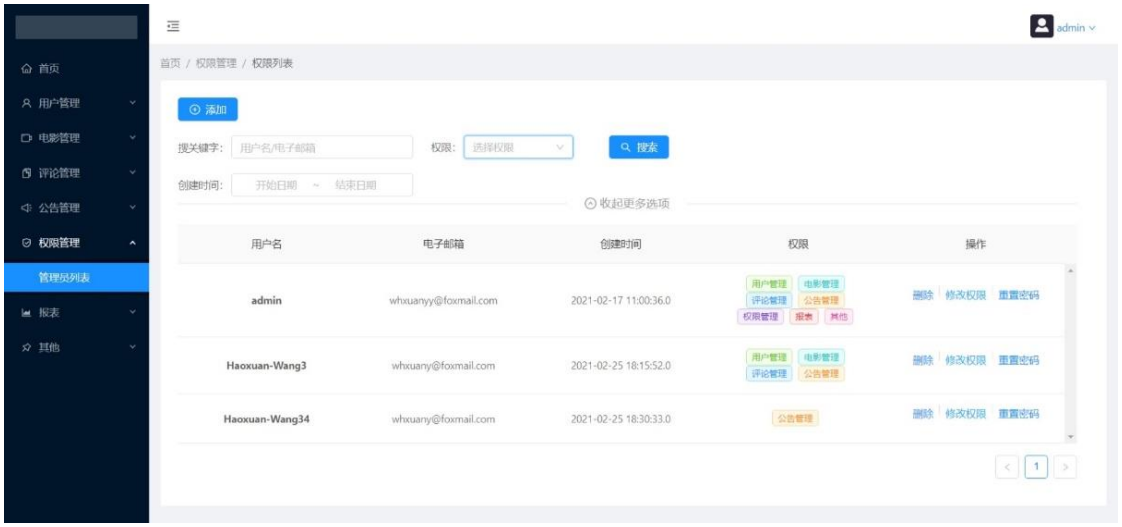

图 4-44 管理员信息管理

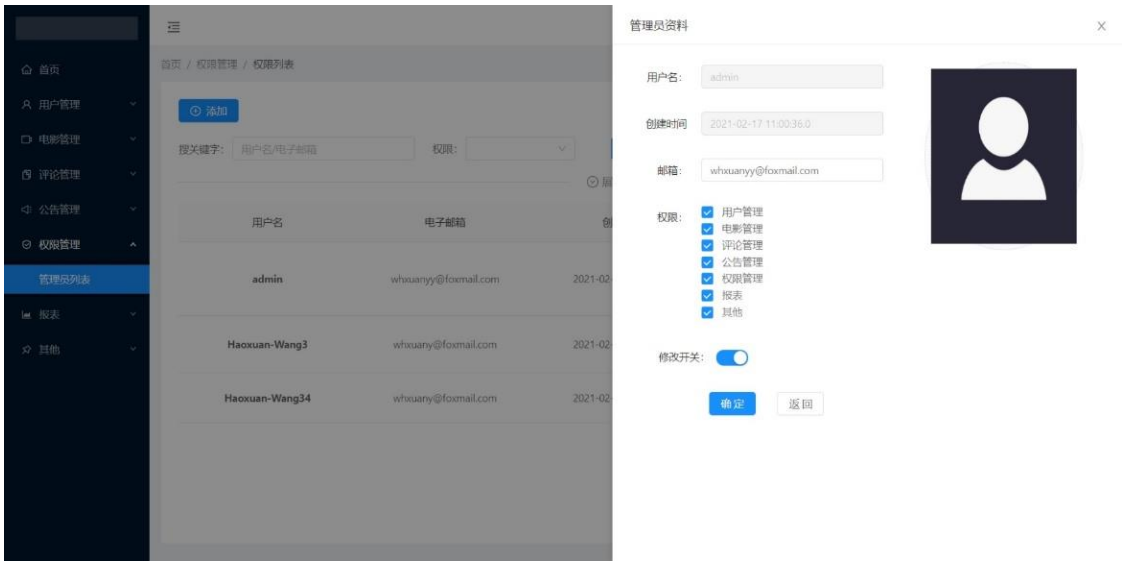

图 4-45 管理员信息修改-权限管理

(6) 轮播图管理

轮播图管理如图 4-46 所示，轮播图上限为 4 张。鼠标移至上传好的图片可选择大图预览和删除操作，删除后恢复空位，可重新上传。未上传的空位，点击上传按钮可从本地上传图片。

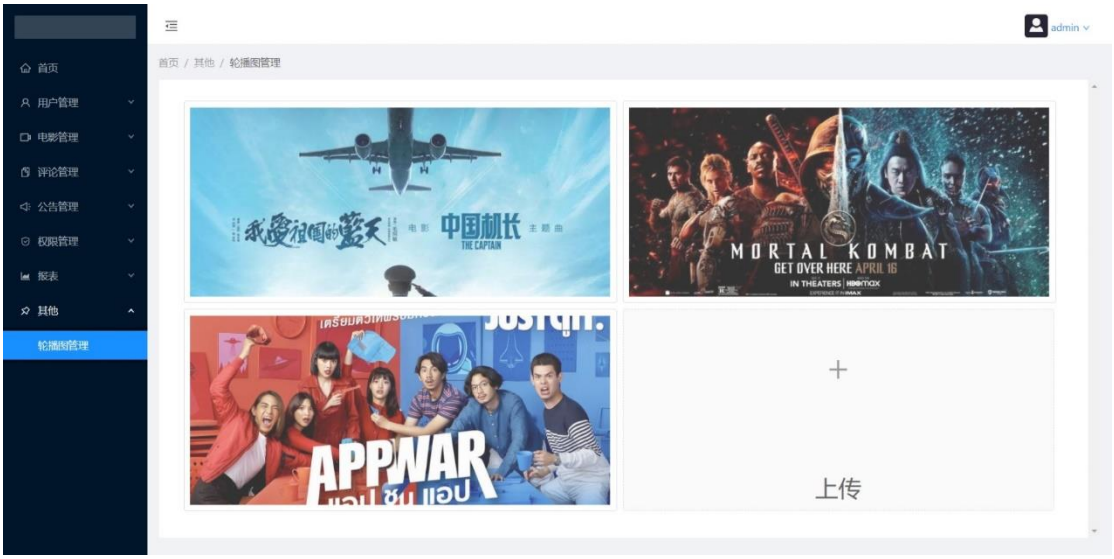

图 4-46 轮播图管理

(7) 数据分析报表

数据分析报表如图 4-47~图 4-51 所示，包含 Top20 实时活跃用户条形图，Top8 实时活跃电影条形图，电影推荐系统用户性别分布饼图、年龄分布柱状图、职业分布饼图。该功能使用前端数据可视化图标库 AntV G2 和 G2Plot 实现。

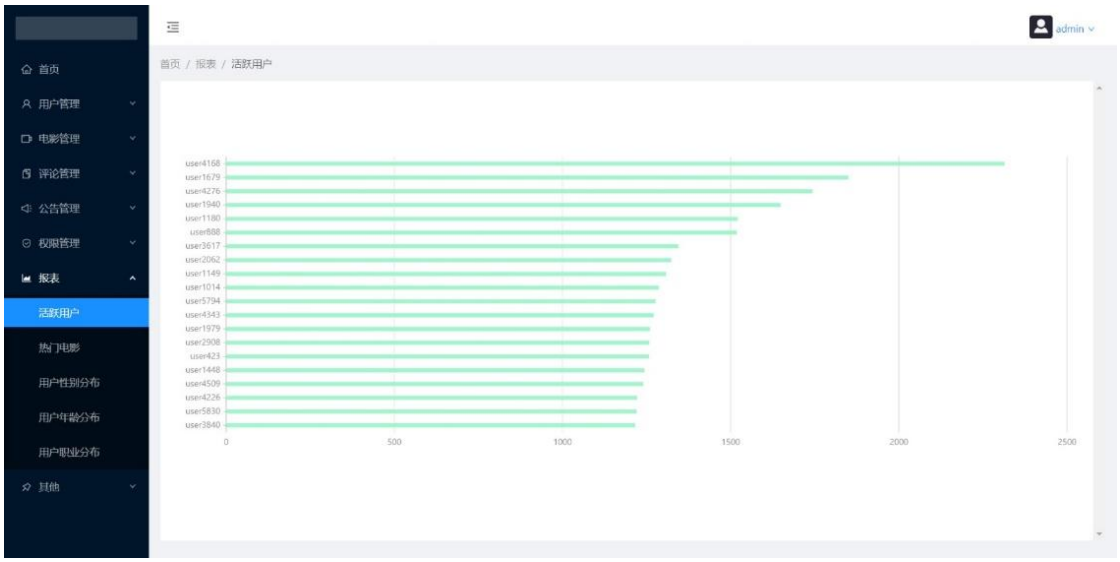

图 4-47 Top20 实时活跃用户条形图

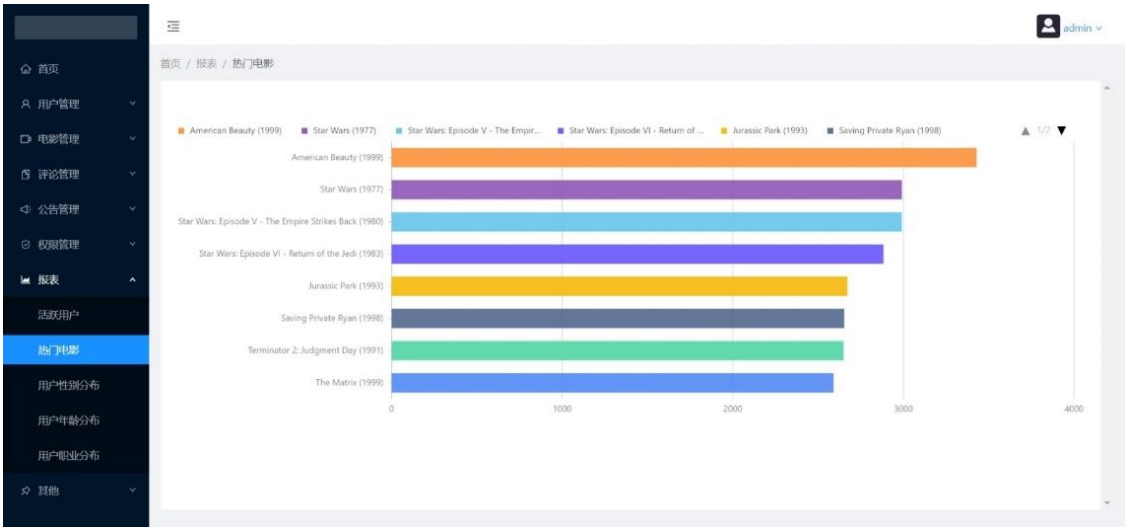

图 4-48 Top8 实时热门电影条形图

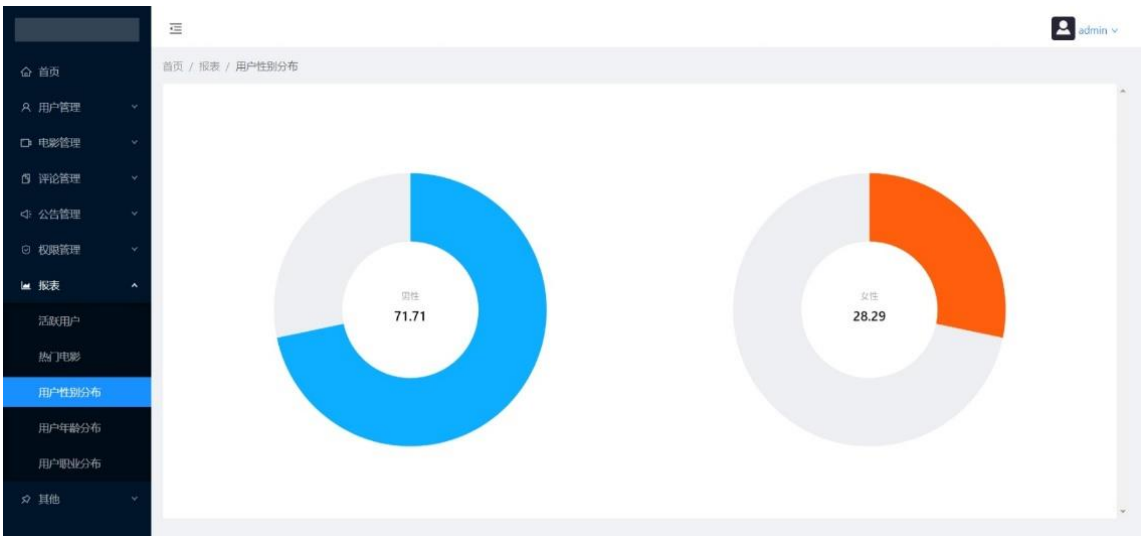

图 4-49 用户性别分布饼图

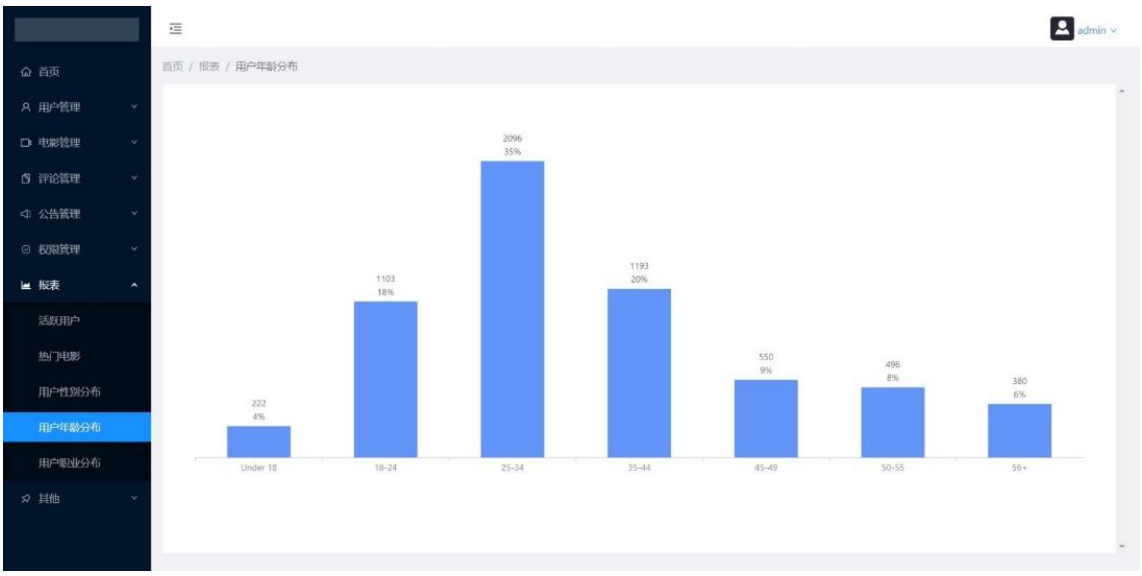

图 4-50 用户年龄分布柱状图

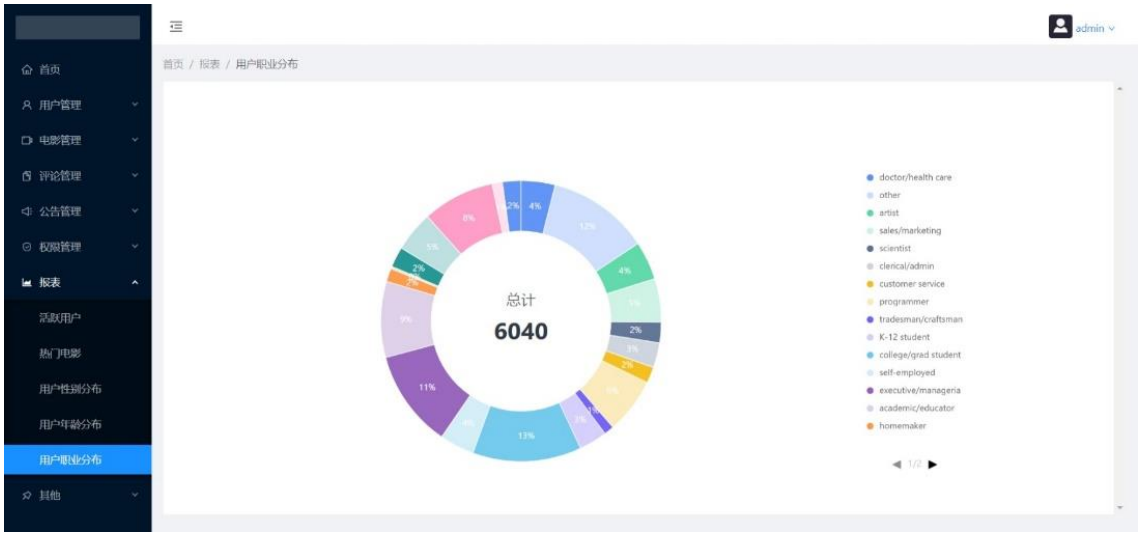

图 4-51 用户职业分布饼图

4.4 系统测试

本节将对本电影推荐系统进行主要业务功能的测试，包含登录注册、用户信息维护、电影信息维护、通知公告管理、电影评论管理、管理端权限维护、电影推荐功能、用户电影检索、历史浏览。

（1）登录测试

表 4-29 登录页面测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	输入框空白，点击登录	无	提示用户名/密码为空	提示用户名/密码为空	Pass
2	勾选管理员登录，输入合法的账号、密码	用户名：admin 密码：123	提示管理员登录成功，并跳转到后台管管理端首页。	提示管理员登入成功，并转到后台管管理端首页。	Pass
3	登录已注册的用	用户名：whx	提示用户登录成	提示用户登录	Pass

	户	密码：123	功，并跳转到用户端首页	成功，并跳转到用户端首页	

经过测试，用户、管理员在进入系统之前，需要输入合法的用户名、密码。经过系统后台将自动进行信息验证，正确即可进入系统，否则友好弹出错误提示。实际功能达到预期，通过。

（2）用户信息维护功能测试

表 4-30 用户信息维护测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	点击新增用户/注册按钮	用户名：newuser 密码：123 性别：男性 年龄段：18-24 职业：other 邮政编码：000000	新增 newuser 用户，并存入数据库	成功新增 newuser 用户，并存入数据库	Pass
2	点击查看 newuser 用户详情，打开修改开关，修改其信息	职业：artist 邮政编码：519000	newuser 用户的职业变为 artist，邮政编码变为 519000	newuser 用户的职业变为 artist，邮政编码变为 519000，并提醒 newuser 用户信息修改成功	Pass
3	在搜索区域通过组合条件，搜索一个用户	输入关键词 user，选择性别男	显示用户名为 user 的男性用户	显示用户名为 user 的男性用户	Pass
4	点击删除 newuser 用户	确认删除	提示 newuser 用户已被删除	提示用户已被正确删除	Pass

经过测试，管理员能够正常删除、新增、修改、检索用户信息。达到预期，通过。

（3）电影信息维护功能测试

表 4-31 电影信息维护测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	点击新增电影按钮	电影名：newmovie 导演：David 编剧：Mitton 演员：Tom/Jame 播放时长：2h 发行时间：2021-1-1 类别：动作/冒险 介绍：This Movie……	新增 newmovie 电影，并存入数据库	成功新增 newmovie 电影，并存入数据库	Pass
2	点击查看电影详情，打开修改开关，修改 newmovie 电影信息	播放时长：2.5h 发行时间：2021-3-5	newmovie 的播放时长修改为 2.5h，发行时间修改为 2021-3-5	newmovie 的播放时长修改为 2.5h，发行时间修改为 2021-3-5,并提	Pass

				醒电影信息修改成功	
3	在搜索区域通过组合条件，搜索一部电影	输入关键词 toy，类别选择喜剧	显示玩具总动员 3 部电影	显示玩具总动员 3 部电影	Pass
4	点击删除 newmovie	确认删除	提示 newmovie 被删除	提示框提醒电影 newmovie 已被删除	Pass

经过测试，管理员能够正常删除、新增、修改、检索电影信息，达到预期，通过。

（4）用户通知公告管理功能测试

表 4-32 用户通知公告管理测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	选择 user1，点击发布公告	公告内容：欢迎光临	新增一条用户通知公告，并存入数据库，用户在首页未读通知信息加一。	成功新增一条用户通知公告，并存入数据库，用户能访问相应未读通知。	Pass
2	修改用例 1 新增的用户通知公告内容	公告内容：欢迎光临本网站	该通知公告信息的信息更新为“欢迎光临本网站”	该通知公告信息的信息更新为“欢迎光临本网站”，并提醒一条用户通知公告修改成功	Pass
3	在搜索区域通过组合条件，搜索一条用户通知公告	输入关键词：变形金刚	显示通知公告“变形金刚 6 上映了……”	显示通知公告“变形金刚 6 上映了……”	Pass
4	删除用例 1 新增的用户通知公告	点击删除通知公告，并确认	该通知公告信息被删除	提示框提醒通知公告信息已被删除	Pass

经过测试，用户能在用户端查看公告，管理员能够对用户通知公告进行维护操作，达到预期，通过。

（5）评论管理功能测试

表 4-33 评论管理测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	用户在 ToyStory1 电影详细页输入影评	电影评论：剧情跌宕起伏，很不错，值得一看！	ToyStory1 电影详细页显示新电影评论“剧情跌宕起伏，很不错，值得一看！”	ToyStory1 电影详细页显示新电影评论“剧情跌宕起伏，很不错，值得一看！”	Pass
2	选择电影	电影评论：留下你	在玩具总动员 1、	在玩具总动员	Pass

	ToyStory1、ToyStory2、ToyStory3，批量新增评论	的评论吧	2、3 详细页中，新增了多条评论信息“留下你的评论吧”，用户在相应电影详细页能看到评论。	1、2、3 详细页中，新增了多条评论信息“留下你的评论吧”，用户在相应电影详细页能看到多条新评论。	
3	修改 ToyStory1 电影评论“留下你的评论吧”	电影评论：欢迎观看本电影	该评论信息变为“欢迎观看本电影”	该评论信息变为“欢迎观看本电影”，并提醒评论信息修改成功	Pass
4	在搜索区域通过组合条件，搜索一条电影评论	输入关键词“剧情跌宕起伏”，评论时间“2021-3-1 至 2021-3-18”	显示评论信息“剧情跌宕起伏，很不错，值得一看！”	显示评论信息“剧情跌宕起伏，很不错，值得一看！”	Pass

经过测试，用户能在用户端添加评论，管理员能够对评论信息进行维护操作。实际效果达到预期，通过。

（6）权限管理功能测试

表 4-34 评论管理测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	点击添加管理员按钮	用户名：admin123 电子邮箱：123@qq.com 权限：用户管理	新增 admin123 管理员，数据存入数据库	成功新增 admin123 管理员，数据库新增数据无误	Pass
2	点击查看管理员详情，打开修改开关，修改 admin123 管理员的权限	赋予操作电影信息管理的权限	该管理员的权限信息发生变化，并能进行相应操作	该管理员的权限信息发生变化，并能进行相应操作	Pass
3	在搜索区域通过组合条件，搜索管理员	选择电影信息管理权	显示具有电影信息管理权限的管理列表	具有电影信息管理权限的管理列表显示正常	Pass
4	删除 admin123 管理员	点击删除 admin123 管理员，并确认	admin123 管理员信息被删除	admin123 管理员信息被删除，并提醒管理员已被删除	Pass

经过测试，权限管理功能，能正常维护管理员数据，能保证不同管理员登录后操作不同界面，防止越权违规操作。达到预期，通过。

（7）电影推荐功能测试

表 4-35 电影推荐测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	登录不同用户账号	用户：user1 密码：123 用户：user2 密码：123	进入网站，并显示不同的推荐列表	进入网站，并推荐列表显示正常，且互不相同。	Pass

电影推荐模块，本文在第 3 章做了丰富的实验，测试结果显示，模型调用正常，且能迅速相应，给予个性且多样的电影推荐，通过。

（8）用户电影检索

表 4-36 用户电影检索测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	登录用户账号，输入关键字，进行关键字检索	关键字：Toy Story	显示玩具总动员 1、2、3 部	搜索页面正常显示三部玩具总动员电影。	Pass
2	登录用户账号，进入电影库，进行分类目录检索。	地区选择欧美，时间选择 90 年代，类型选择动作。	仅显示 90 年代欧美的动作片	电影库显示正常，均为 90 年代欧美动作片	Pass

经过测试，用户能通过关键字、分类目录检索电影。实际效果达到预期，通过。

（9）历史记录浏览

表 4-37 历史记录浏览测试用例表

用例编号	测试步骤	输入数据	预期结果	实际结果	测试结果
1	用户访问不同的电影详细页	用户分别按顺序访问 ToyStory1、ToyStory2 后，点击浏览记录	浏览记录先后显示 ToyStory2 、ToyStory1	浏览记录正常显示，且顺序正常。	Pass
2	在执行用例 1 后，用户继续访问不同的电影详细页	用户接着分别按顺序访问 ToyStory3、ToyStory1 后，点击浏览记录	浏览记录先后显示 ToyStory1 、ToyStory3 、ToyStory2。	浏览记录正常显示，且顺序正常。	Pass
3	用户点击删除历史记录	删除 ToyStory2	浏览记录更新显示，现先后显示 ToyStory1 、ToyStory3	浏览记录删除正常，且顺序正常。	Pass
4	用户点击清空历史记录按钮	无	浏览记录清空	浏览记录清空正常	Pass

经过测试，历史记录浏览功能，能正常显示、正常删除与清空历史浏览。实际效果达到预期，通过。

4.5 本章小结

本章从需求分析出发，研究与设计了电影推荐系统的系统架构，采用 SpringBoot+Vue 前后端分离的方式进行开发。在智能推荐模块，训练及部署了相应的深度学习模型，实现了个性化推荐与基于个性化精排序的实时热门推荐，并对系统进行实现展示和测试，实际效果达到预期。 本文设计的电影推荐系统健壮、高效、稳定，推荐功能准确、实时、个性且多样，具有良好的用户体验。

5 总结与展望

5.1 全文总结

本文的所做的工作总结如下：

（1）本文首先介绍了推荐系统的研究背景及价值。确立了将基于深度学习的协同过滤作为本文的研究方向。

（2）本文对基于协同过滤的经典推荐算法进行了介绍，接着引出了基于深度学习的协同算法。复现了 GMF、NCF、ConvNCF、ENMF 等前沿的深度协同推荐算法，并将 sketch-based 的大规模流式数据处理技术与深度推荐模型混合，提出基于个性化精排序的热门推荐方法，极大提升了传统热门推荐的性能。另外，本文还对上述模型进行了丰富的对比实验，详细分析了超参数对模型的影响。

（3）本文对电影推荐系统进行了详细的需求分析，并以此为依据，研究与设计了整个电影推荐系统。在表现层，本文使用了 Vue 脚手架，结合当前热门的 Ant-design-Vue、AntV G2/G2Plot 组件库进行前端页面开发。在业务层，本文使用 SpringBoot 进行后端开发。系统开发基本上满足前后端分离。在算法层，本文将所研究的深度协同推荐模型进行工程化。实现了基于 ENMF 的个性化深度协同推荐和基于 MV-sketch 与 NCF 的个性化实时热门推荐。在数据层，本文采用了 MySQL 和 MongoDB 进行大数据的存取与处理。通过系统的测试，本文设计与实现的电影推荐系统健壮、高效、稳定，核心推荐功能准确、实时、满足个性化且具有推荐多样性。

5.2 后续工作展望

推荐系统是一个庞大的课题，可归属于信息检索领域，但具有强大的应用背景，涉及机器学习、深度学习、计算机视觉、自然语言处理、工程学等多方面知识。由于现阶段条件有限，只能做到片面的一点研究。在后续的方面，将继续围绕以下内容进行研究与改进：

（1）深度推荐模型嵌入层。在深度推荐模型中，嵌入是不可缺少的基本操作。嵌入层完成了高维稀疏特征的降维操作，降维后的向量作为模型输入的必不可少的一部分，同时也可以反馈向量间的特性，如相似度。目前，相关推荐嵌入的研究也比较少，本文的模型实现所使用的嵌入操作是基于 Pytorch 框架 nn 包中的词嵌入方法。在后续的工作中，将对其进行研究，考虑将自然语言处理嵌入层的最新成果引入推荐系统。另外，本文的系统实现由于硬件环境限制，仅使用了用户 ID 和电影 ID 进行嵌入。在条件允许的情况下，后续将引入更多特征进行嵌入，以提升推荐性能。

（2）推荐系统的隐私保护。由于推荐需要使用用户的高维特征，在享受推荐带来的便利时，用户也伴随着个人信息泄露的风险。目前，推荐系统的隐私保护是各大互联网公司及推荐系统专家的一大研究热点。后续将对其进行研究，以改进升级本文实现的系统。

（3）大规模流数据处理框架。出于硬件环境原因，目前本系统没有使用大数据处理框架，无法高并发的处理大规模流式数据。后续考虑使用 Spark、Strom、Flume 等大数据处理框架对系统进行改进升级，以应对高并发的大规模用户访问流。

致谢

随着论文编写进入尾声，本科生涯即将介绍。站在新征程的起点，回顾四年，满满都是老师和同学的鼓励与帮助。

首先，我要感谢肖老师对我的指导。我从大二就进入了肖老师的课题组。每次开会讨论，他都会无形地教导我们做事要认真、严谨，要如何读文献、如何确定主题与找出问题。在课题组进行科研训练，不仅丰富了自己的学科素养，为研究生阶段打下了良好的基础，而且培养了一种乐观积极的生活态度和坚强的意志。这对我解决当前或未来学习生活上的难题，是有很大帮助的。能遇到肖老师我感到十分幸运，他治学严谨、认真负责的工作态度对我有很大影响。在此，我真心祝愿恩师身体健康，工作顺利！

感谢福建师范大学数学与信息学院对我的栽培！同时，我也要感谢大学四年以来，所有的课任老师们以及辅导员们。感谢他们的辛勤付出，细心教授。

我还要感谢我亲爱的家人们，有他们的支持和鼓励，是我学习的动力。

我也要感谢课题组的师哥师姐与同学们对我的帮助和支持。

最后，我衷心地感谢各位参与评审论文的专家老师们，谢谢你们百忙之中抽出时间来评审、指导我的论文，谢谢！

参考文献

[1] 项亮.推荐系统实践[M].人民邮电出版社,2012.
[2] 王茜子.基于混合算法的电影推荐系统的研究与设计[D].电子科技大学,2020.
[3] Netflix,Netflix prize[EB/OL].https://www.netflixprize.com/index.html,2009-09-21/2021-03-18.
[4] Karatzoglou A,Hidasi B.Deep Learning for Recommender Systems[C]// the Eleventh ACM Conference.ACM,2017.
[5] He X,Liao L,Zhang H et al.Neural Collaborative Filtering[J].International World Wide Web Conferences Steering Committee,2017.
[6] Chen C,Zhang M,Zhang Y et al.Efficient Neural Matrix Factorization without Sampling for Recommendation[J].ACM Transactions on Information Systems,2020,38(2):1-28.
[7] Chen C,Zhang M ,Wang C et al.An Efficient Adaptive Transfer Neural Network for Social-aware Recommendation[C]// the 42nd International ACM SIGIR Conference.ACM,2019.
[8] Chong Chen,Min Zhang,Weizhi Ma et al.Efficient Non-Sampling Factorization Machines for Optimal Context-Aware Recommendation[C].In Proceedings of The Web Conference 2020 (WWW'20),2020,2400–2410.
[9] Chen C,Zhang M,Ma W et al.Jointly Non-Sampling Learning for Knowledge Graph Enhanced Recommendation[C]//SIGIR '20: The 43rd International ACM SIGIR conference on research and development in Information Retrieval.ACM,2020.
[10] Chen,ZhangM.et al.Efficient Heterogeneous Collaborative Filtering without Negative Sampling for Recommendation[C].AAAI,2020.
[11] He,Ruining and Julian McAuley.VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback[C].AAAI,2016.
[12] Xin Dong,Lei Yu,Zhonghuo Wu et al.A hybrid collaborative filtering model with deep structure for recommender systems[C].In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17).AAAI Press,2017,1309–1315.
[13] Breese J S,David Heckerman,Kadie C.Empirical Analysis of Predictive Algorithms for Collaborative Filtering[J].Uncertainty in Artificial Intelligence,2013,98(7):43-52.
[14] A.Töscher and M.Jahrer.The BigChaos solution to the Netflix Prize 2008[R].Technical report,commendo research & consulting,October 2008.
[15] Paterek, A.Improving regularized singular value decomposition for collaborative filtering[C].Proceedings of KDD Cup and Workshop, 2007.

[16] Koren Y.Factor in the neighbors: Scalable and accurate collaborative filtering[J]. Acm Transactions on Knowledge Discovery from Data,2010,4(1):1-24.
[17] Xiangnan He, Xiaoyu Du, Xiang Wang et al. Outer Product-based Neural Collaborative Filtering[C].IJCAI 2018: 2227-2233.
[18] HengTze Cheng,Levent Koc,Jeremiah Harmsen et al.Wide & Deep Learning for Recommender Systems[C]. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems(DLRS 2016),2016,7–10.
[19] Ruoxi Wang, Bin Fu, Gang Fu et al.Deep & Cross Network for Ad Click Predictions[C].In Proceedings of the ADKDD'17(ADKDD'17),2017,Article 12,1–7.
[20] Ioffe, Sergey and C.S.Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift[C].Proceedings of The 32nd International Conference on Machine Learning,2015,pp.448–456.
[21] Hadji,I. and Wildes et al.What Do We Understand About Convolutional Networks?[J/OL]. ArXiv e-prints,https://arxiv.org/abs/1803.08834,2018.
[22] Krizhevsky A,Sutskever I,Hinton G.ImageNet Classification with Deep Convolutional Neural Networks[C]// NIPS. Curran Associates Inc. 2012.
[23] Ruder, S. overview of gradient descent optimization algorithms[J/OL].ArXiv e-prints, https://arxiv.org/abs/1609.04747,2016.
[24] Grouplens, MovieLens 1M Dataset[DB/OL].https://grouplens.org/datasets/movielens/1m. 2015-9-23/2021-3-18.
[25] Rendle S,Freudenthaler C,Gantner Z et al.BPR: Bayesian personalized ranking from implicit feedback[C]// UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. AUAI Press, 2009.
[26] Hu Y,Koren Y,Volinsky C.Collaborative Filtering for Implicit Feedback Datasets[C]// Eighth IEEE International Conference on Data Mining.IEEE,2009.
[27] Cormode G,Muthukrishnan S.An Improved Data Stream Summary:The Count-Min Sketch and Its Applications[J].Journal of Algorithms, 2004, 55(1):58-75.
[28] Moses Charikar,K. C. B,F. C. C et al. Finding Frequent Items in Data Streams[J]. Theoretical Computer Science,2002,312(1):3-15.
[29] Tang L,Huang Q,Lee P.MV-Sketch: A Fast and Compact Invertible Sketch for Heavy Flow Detection in Network Data Streams[C]//IEEE INFOCOM 2019 -IEEE Conference on Compute Communications.IEEE,2019.
[30] Boyer R S, Moore J S. MJRTY—A Fast Majority Vote Algorithm[J].Springer Netherlands, 1991.

Research and Implementation of Movie Recommender System Based on Deep Learning

School of Mathematics and Information Technology Software Engineering
116052017051 Hao-Xuan Wang Tutor: Ru-Liang Xiao

Abstract: Personalized recommendation not only enhances users' experience of using the system, but also brings great economic benefits to service providers, which has significant research value. In the era of big data, traditional collaborative recommendation algorithms cannot bring well user experience due to the problems of cold start and ranking loss. In this paper, we have fully researched and experimented with the state-of-the-art of deep recommendation models, and designed and implemented a hybrid movie recommender system with personalized recommendation and popular recommendation. In the personalized recommendation module, the ENMF model with the best experimental performance is chosen. In the popular recommendation module, this paper mixes sketch-based frequent item statistics technique for large-scale streaming data and deep recommendation model, and proposes a real-time popular movie recommendation method based on personalized fine ranking, and chooses the ItemPop-NCF model with the best experimental performance. This paper complements the MovieLens-1m dataset by crawlers, and after testing, the recommendation function of this hybrid framework system is accurate, personalized and diverse, and the information processing is real-time, stable and efficient.

Keywords: Personalized Recommendation; Popular Recommendation; Hybrid Recommender Framework; Deep Learning