

**本科生毕业论文**



**题 目 中文电影评论情感分析系统设计与实现**

**学 院**

**专 业**

**学生姓名**

**学 号**  **年级 2021级**

**指导教师**

**教务处制表**

**二〇二五 年 五 月 二十 日**

**中文电影评论情感分析系统设计与实现**

学生 指导老师

**摘要：**随着互联网的普及，在线电影评论已成为观众表达观影体验和意见的重要途径,同时也为电影行业从业人员提供了进行市场分析、行业规划和关注行为研究的重要数据来源。电影评论情感分析任务的目的是从情感表达角度对评论进行多类别分类，有助于行业人员更准确观众情绪变化，调整营销策略，并为观众提供个性化影评推荐。

为此，本文基于Bidirectional Encoder Representations from Transformers（BERT）模型设计并实现了一个中文电影评论情感分析系统。该系统实现了用户管理、电影管理、评论管理、情感分析、数据分析和系统管理六大功能模块，包括了用户注册，用户登录，电影信息展示，电影搜索，评论发表，评论情感分析，情感趋势分析等二十个子功能。通过爬取视频网站数据，本文构建了一个包含1018部电影与4324956条影评的中文电影评论数据集，并根据评论所属影片及情感类别进行标注与整理。在该数据集上对预训练BERT模型进行微调，实现了对电影评论的三分类情感分析任务。实验结果表明微调后的模型对中文电影评论情感分析准确率达到了90%以上，能够有效识别影评中的情感倾向。本系统在电影推荐、观众反馈分析和市场趋势预测等场景中具有广泛的应用潜力。

**主题词：**电影评论；情感分析；BERT

**Design and Implementation of Sentiment Analysis System for Chinese Movie Reviews**

Cybersecurity

Student： Zhiyang Fang Adviser: □□□

**Abstract：**With the popularization of the Internet, online movie reviews have become an important way for viewers to express their movie watching experience and opinions, and at the same time, they also provide an important source of data for movie industry practitioners to conduct market analysis, industry planning, and research on attention behaviors. The purpose of the movie review sentiment analysis task is to classify multiple categories of reviews from the perspective of sentiment expression, which can help industry practitioners to more accurately analyze the audience's sentiment changes, adjust marketing strategies, and provide personalized movie review recommendations for the audience.

To this end, this paper designs and implements a Chinese movie review sentiment analysis system based on the Bidirectional Encoder Representations from Transformers (BERT) model. The system realizes six functional modules: user management, movie management, review management, sentiment analysis, data analysis and system management, including twenty sub-functions, such as user registration, user login, movie information display, movie search, review publication, review sentiment analysis, and sentiment trend analysis. By crawling the data from video websites, this paper constructs a Chinese movie review dataset containing 1018 movies and 4324956 movie reviews, and annotates and organizes the reviews according to the film and sentiment categories they belong to. The pre-trained BERT model is fine-tuned on this dataset to realize the three-classification sentiment analysis task for movie reviews. The experimental results show that the fine-tuned model achieves an accuracy of more than 90% for the sentiment analysis of Chinese movie reviews, and is able to effectively identify the sentiment tendencies in movie reviews. This system has a wide range of potential applications in movie recommendation, audience feedback analysis and market trend prediction scenarios.

**Key Words：**Movie Reviews; Sentiment Analysis; BERT

**目 录**

[1 绪论 1](#_Toc196466535)

[1.1 研究背景 1](#_Toc196466536)

[1.2 国内外研究现状 2](#_Toc196466537)

[1.3 论文主要工作 3](#_Toc196466538)

[1.4 论文组织与结构 3](#_Toc196466539)

[2 相关技术 5](#_Toc196466540)

[2.1 情感分析 5](#_Toc196466541)

[2.2 BERT 5](#_Toc196466542)

[2.3 MongoDB 6](#_Toc196466543)

[2.4 本章小结 6](#_Toc196466544)

[3 系统需求分析与概要设计 8](#_Toc196466545)

[3.1 系统需求分析 8](#_Toc196466546)

[3.1.1 系统需求概述 8](#_Toc196466547)

[3.1.2 需求清单 8](#_Toc196466548)

[3.1.3 系统数据分析 9](#_Toc196466549)

[3.1.4 系统功能分析 9](#_Toc196466550)

[3.2 系统概要设计 10](#_Toc196466551)

[3.2.1 系统设计概述 10](#_Toc196466552)

[3.2.2 系统数据库设计 11](#_Toc196466553)

[3.2.3 系统用户界面设计 13](#_Toc196466554)

[3.2.4 系统API接口设计 14](#_Toc196466555)

[3.3 本章小结 15](#_Toc196466556)

[4 系统详细设计 16](#_Toc196466557)

[4.1 实现环境与工具 16](#_Toc196466558)

[4.1.1 硬件环境 16](#_Toc196466559)

[4.1.2 实现工具 16](#_Toc196466560)

[4.2 用户登录模块 16](#_Toc196466561)

[4.2.1 用户认证实现 17](#_Toc196466562)

[4.2.2 密码加密实现 17](#_Toc196466563)

[4.3 电影展示模块 19](#_Toc196466564)

[4.3.1 电影数据集爬取 19](#_Toc196466565)

[4.3.2 数据清洗 21](#_Toc196466566)

[4.4 电影评论分析模块 24](#_Toc196466567)

[4.4.1 电影评论数据爬取 24](#_Toc196466568)

[4.4.2 数据格式转换 24](#_Toc196466569)

[4.4.3 评论数据可视化实现 25](#_Toc196466570)

[4.5 情感分析模块 26](#_Toc196466571)

[4.5.1 情感分析的模型框架 26](#_Toc196466572)

[4.5.2 模型的训练 29](#_Toc196466573)

[4.5.3 情感分析结果可视化 33](#_Toc196466574)

[4.6 用户访问控制实现 36](#_Toc196466575)

[4.6.1 管理员访问控制实现 36](#_Toc196466576)

[4.6.2 普通用户访问控制实现 36](#_Toc196466577)

[4.7 评论管理功能实现 37](#_Toc196466578)

[4.8 数据分析功能实现 39](#_Toc196466579)

[4.9 本章小结 40](#_Toc196466580)

[5 系统测试 41](#_Toc196466581)

[5.1 系统部署与优化 41](#_Toc196466582)

[5.2 性能优化与负载测试 41](#_Toc196466583)

[5.3 用户功能模块测试 41](#_Toc196466584)

[5.4 电影模块测试 42](#_Toc196466585)

[5.5 评论管理模块测试 42](#_Toc196466586)

[5.6 情感分析模块测试 42](#_Toc196466587)

[5.7 数据分析模块测试 43](#_Toc196466588)

[5.8 系统管理模块测试 43](#_Toc196466589)

[5.9 本章小结 43](#_Toc196466590)

[6 总结与展望 44](#_Toc196466591)

[6.1 总结 44](#_Toc196466592)

[6.2 展望 44](#_Toc196466593)

[参考文献 45](#_Toc196466594)

[致 谢 48](#_Toc196466595)

# 绪论

* 1. 研究背景

随着互联网的不断普及，大众在网络世界越来越活跃从而发表了大量的言论，这些言论表达了大众的想法和情感，通过这些言论，我们可以捕捉到大众的想要表达的相关信息，同时随着国民生活水平不断提高，人民对美好生活的向往愈加浓烈。娱乐方式呈现出多样化趋势，其中走进电影院观影成了这其中必不可少的一种娱乐方式[1]。那么电影影评已经成为观众表达观影体验的重要渠道[2][3]，如图1-1所示，大众对电影发表了大量的影评，这些影评不仅包含了大众对电影的客观评价还包含了大众的主观情感，往往这些情感的表达才反应了大众内心最真实的想法，所以如何利用以及利用好这些藏在中文电影影评中的情感就显得尤为必要。



图1-1 影评数据

如何创作出真正吸引观众的高质量电影，一直是困扰许多电影创作者的重要课题；与此同时，面对数量庞大的院线影片，观众如何快速做出观影选择，也日益成为实际需求。在这一背景下，评论文本作为观众反馈的重要载体，蕴含着丰富的情感信息，具有重要的应用价值。基于人工方式进行评论分析不仅效率低下，且难以适应大规模数据处理的要求。为此，构建一套能够自动化识别评论情感倾向的分析系统，不仅能够辅助创作者优化内容、帮助观众做出个性化选择，也为电影行业提供了智能化的数据支撑，已成为自然语言处理技术的重要应用方向之一[4][5]。

本项目设计并实现了一套中文电影评论情感分析系统，具备评论数据的自动采集、文本预处理、情感分类和结果展示等核心功能。系统能够从主流平台抓取用户评论，利用自然语言处理技术对文本进行清洗与分词，并通过训练好的分类模型自动识别评论的情感倾向（如正面、负面、中性）。分析结果以可视化形式呈现，支持用户按影片或关键词查看评论情绪分布。该系统可用于辅助行业人员快速获取观众反馈，优化影片营销策略，同时为观众提供更具参考价值的观影决策依据。

* 1. 国内外研究现状

情感分析作为自然语言处理的重要分支，近年来在技术、应用和研究层面均取得了显著进展。以BERT、GPT等为代表的预训练语言模型已成为主流工具，大幅提升了情感识别的准确率，而多模态分析、细粒度方面级情感分析（ABSA）等技术进一步拓展了其应用场景。目前，许多评论分析系统利用自然语言处理技术进行评论情感分析，以实现自动化的检测大众的情绪及意见。其中，利用预训练语言模型进行文本情感分析是比较先进的技术，可以高效地实现自动化的评论理解和分类。

国外早期的情感分析研究主要采用监督学习方法和情感词典方法。Pang等人首次将机器学习应用于电影评论情感分类，比较了SVM、朴素贝叶斯和最大熵模型在二元情感分类（正面/负面）上的表现。研究发现，使用unigram特征时SVM表现最佳（准确率约82.9%），为后续文本分类任务奠定了基础[6]。在词典方法方面，Taboada等人提出了基于SentiWordNet的词典方法，通过计算文本中情感词的极性得分（正/负/中性）来预测整体情感。该方法在商品评论分析中达到约74%的准确率，但面临否定词处理（如"not good"）和上下文依赖的挑战[7]。

随着深度学习技术的发展，神经网络在情感分析领域展现出更强的建模能力和更高的准确率。Kim等人提出使用静态词向量（Word2Vec）和卷积神经网络（CNN）提取文本局部特征。在MR电影评论数据集上，单通道CNN达到81.5%的准确率，多通道CNN提升至82.7%[8]。Tang 等人设计了基于长短期记忆网络（LSTM）的模型，通过建模词序依赖关系提升长文本分析效果。在Stanford Sentiment Treebank数据集上，LSTM比CNN的准确率提高约3%[9]。Vaswani等人提出的Transformer架构通过自注意力机制（Self-Attention）动态计算词间权重，在机器翻译任务中取得突破，为后续BERT等模型奠定基础[10]。

近年来，随着预训练语言模型的发展，情感分析任务迎来了质的飞跃。Devlin 等人出的双向Transformer模型通过掩码语言建模（MLM）学习上下文相关表示。在SST-2情感数据集上，BERT-base达到92.7%的准确率，较LSTM提升约10%[11]。Zadeh等人构建的CMU-MOSEI数据集包含23,500条YouTube视频片段，涵盖文本、音频和视觉模态。实验表明，多模态融合模型（如TFN）比单模态文本分析的F1值提高15%[12]。

总结来说，各种自然语言处理技术为情感分析系统的发展提供了基础。当前，情感分析已广泛应用于商业营销、金融舆情、心理健康和人机交互等领域，例如产品评论分析、社交媒体情绪监测和智能助手优化。同时这些技术的综合应用使得自动化评估电影影评成为可能。随着机器学习和自然语言处理领域的进一步发展，情感分析系统的性能和准确性将得到持续提高。与此同时，中文语料和多模态分析也将在未来的研究中得到更多的关注。

* 1. 论文主要工作

本论文的核心工作是构建并实现一个面向中文电影评论的情感分析系统。系统基于预训练模型 BERT-base-Chinese，通过微调使其具备对评论文本情感倾向的自动识别能力，并通过后端 API 部署模型，实现情感分析功能的在线调用。在此基础上，系统集成多个功能模块，不仅支持情感识别，还具备完善的用户交互界面和后台管理功能，提升了整体使用体验与系统实用性。根据功能设计，系统主要由以下六个模块组成：

（1）普通用户管理模块：支持用户注册、登录与个人信息管理。用户可通过用户名、邮箱及密码完成注册，并在登录后查看和修改个人资料，包括头像与联系方式等。

（2）电影管理模块：提供电影信息的展示功能，包括影片的标题、类型、简介、评分与评论等，用户可在此查看电影的完整信息。

（3）评论管理模块：支持评论的查看、发布、编辑与删除。用户可为电影撰写评论和评分，系统支持分页浏览和按时间或热度排序评论内容。

（4）情感分析模块：通过微调后的 BERT 模型对用户评论进行情感分类（正面/负面/中性），并展示各影片评论的情感分布与变化趋势。

（5）数据分析模块：提供用户行为分析、电影评分分布、影片类型统计等功能，便于系统管理员或运营方了解用户活跃情况与内容偏好。

（6）系统管理模块：用于对电影、评论等核心数据进行统一管理，支持数据审查、权限配置与系统维护等操作。

* 1. 论文组织与结构

本论文由六部分组成，各部分主要内容如下：

第一部分：绪论。本章介绍本项目的研究背景，陈述本项目相关的国内外研究现状，最后对本文的主要工作作出介绍。

第二部分：背景知识简介。主要介绍了项目涉及的相关技术知识背景。

第三部分：系统概要设计。主要介绍了系统设计的概要情况。

第四部分：系统详细设计与实现。主要介绍了系统的详细设计流程和实现的具体方法，对布署微调后的BERT模型以实现对中文电影评论进行情感分析；包括具体功能的实际效果和代码说明。

第五部分：系统测试。对以实现模型的实际应用为目的开发的Web应用进行测试，包括负载测试，功能测试和用户测试。

第六部分：总结和展望。本章写出了项目整个开发过程中的感悟和反思，总结了本篇论文的工作和主要贡献并对现存问题和未来发展发表了自己的看法。

# 相关技术

* 1. 情感分析

情感分析是自然语言处理（NLP）的重要研究方向，其发展经历了从传统方法到深度学习的转变。早期研究主要采用基于词典的方法，例如SentiWordNet[13]和VADER[14]，通过词汇极性匹配进行情感分类，但这类方法难以处理复杂的语义依赖。随着机器学习的发展，支持向量机（SVM）和朴素贝叶斯等算法被引入情感分析任务，并在电影评论数据集上实现了约85%的准确率[15]。

近年来，深度学习技术推动了该领域的研究进展。Kim等人首次将卷积神经网络（CNN）应用于句子级情感分类[16]，而Vaswani等人提出的Transformer架构为后续预训练模型奠定了基础。Devlin等人开发的BERT模型[11]通过双向上下文表征提升了细粒度情感分析的性能，在SST-2等基准数据集上达到最优水平。Radford等人提出的GPT系列模型实现了生成式方法在情感分析中的应用，能够生成自然的情感文本。在应用层面，Liu等人优化的RoBERTa模型[15]被用于社交媒体舆情监测，而Brown等人构建的GPT-3则在跨领域情感理解中表现出有效能力。

当前研究热点涉及多模态情感分析（结合文本、语音和视觉数据）、领域自适应方法，以及解决模型偏见和可解释性等伦理问题。未来研究方向包括结合认知心理学理论的情感计算和面向低资源语言的轻量化模型，这些进展将推动情感分析技术在电商、金融、心理健康等领域的应用。

* 1. BERT

BERT（Bidirectional Encoder Representations from Transformers）是由Devlin等人[11]于2018年提出的突破性预训练语言模型，其核心创新在于采用双向Transformer编码器结构和掩码语言建模（MLM）任务。通过随机遮盖输入文本中15%的词汇并预测被遮盖词，BERT能够同时利用上下文信息学习词的深层语义表征。此外，预训练阶段还包括下一句预测（NSP）任务，以捕捉句子间的逻辑关系。

BERT-base（12层）和BERT-large（24层）在11项NLP处理任务中达到最优性能，其中GLUE基准测试的平均性能提升7.7%[11]。后续研究进一步优化了BERT的训练策略，例如Liu等[15]人提出的RoBERTa通过动态掩码和更大批次训练提升了模型效果。在应用层面，BERT及其变体已广泛应用于问答系统[17]、文本分类[18]和命名实体识别[19]等任务。其双向编码能力显著改善了语言理解的准确性。

然而，BERT的计算资源消耗大[20] ，且在跨领域任务中存在适应性不足的问题[21]。为解决这些问题，研究者探索了知识蒸馏[22]和参数高效微调[23]等方法，以降低模型复杂度并提升泛化能力。当前，基于BERT的架构仍是NLP领域的重要基础，深刻影响了后续大语言模型的发展[24]，BERT模型框架如图2-1所示：

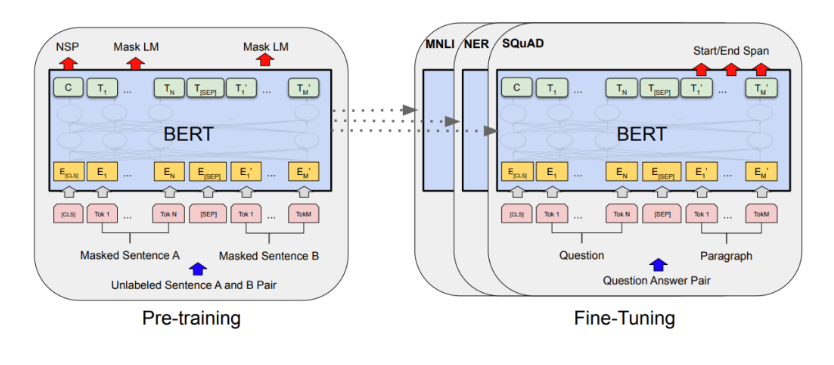


图2-1 BERT模型图

* 1. MongoDB

MongoDB作为文档型NoSQL数据库，采用BSON格式存储数据，其灵活的文档模型支持动态模式和嵌套结构，能够高效管理非结构化及半结构化数据[25]。该数据库通过分片集群架构实现水平扩展，结合WiredTiger存储引擎优化读写性能，同时提供聚合管道功能以支持复杂数据处理与分析[26][27]。在查询优化方面，MongoDB支持单字段、复合、多键及地理空间等多种索引类型，有效提升数据检索效率[28]。安全层面，其内置基于角色的访问控制（RBAC）和传输层加密（TLS）等机制，满足企业级数据保护需求[29]。实际应用中，MongoDB在物联网设备数据存储、内容管理系统及实时分析场景中表现突出，能够应对高并发读写与海量数据存储需求。尽管其在简单事务处理中性能优异，但复杂事务场景仍存在局限性。最新版本通过引入分布式事务支持与增强型聚合操作符，进一步扩展了其在多模态数据管理中的应用能力[30]。

* 1. 本章小结

本章围绕情感分析、BERT模型和MongoDB数据库三个核心主题展开，系统介绍了相关技术的发展历程、核心原理及实际应用。

在情感分析部分，本章回顾了该领域从早期基于词典的方法到现代深度学习技术的演进过程。这些技术进步为电商、金融等领域的精细化情感计算提供了坚实基础。

关于BERT模型，本章详细阐述了其双向Transformer架构和掩码语言建模的创新设计，解释了其在上下文表征学习方面的突破性优势。通过对比BERT-base/large的基准性能及RoBERTa等改进模型，说明了预训练-微调范式对NLP任务的普适性价值。

在MongoDB方面，本章剖析了该文档型数据库的BSON存储格式、分片集群架构等核心技术特性，总结了其在处理非结构化数据时的灵活性和扩展性优势。结合索引优化、安全机制等企业级功能，讨论了该数据库在物联网、实时分析等场景的应用价值，也客观分析了其事务处理能力的现存局限。

# 系统需求分析与概要设计

* 1. 系统需求分析
     1. 系统需求概述

系统最核心的需求是需要通过BERT预训练模型实现对中文电影评论的情感分析。在分析之前，首先需要先通过网络爬虫爬取公开的电影和其评论数据，随后对原始评论数据进行预处理，包括数据清洗与中文分词等操作。如此经过预处理后的评论数据才能够输入 BERT 模型与情感分类器进行情绪倾向分析，输出情感指数。最终系统需要根据输出的分析结果进行可视化展示，并且需要结合电影的基本信息、电影相关内容综合呈现。

* + 1. 需求清单

系统功能需求包括以下六个模块，分别是用户管理模块、电影管理模块、评论管理模块、情感分析模块、数据分析模块、系统管理模块，其功能及其具体描述如表 3‑1所示。

**表 3‑1 需求功能清单**

|  |  |  |
| --- | --- | --- |
| **模块** | **功能** | **功能描述** |
| 用户功能模块 | 用户注册 | 新用户通过填写用户名、邮箱和密码进行注册 |
| 用户登录 | 已注册用户通过用户名和密码进行登录 |
| 个人信息管理 | 用户可以查看和修改个人信息，包括头像、邮箱等 |
| 电影模块 | 电影信息展示 | 展示电影基本信息，包括标题、介绍、类型等 |
| 电影搜索 | 支持按标题、类型等条件搜索电影 |
| 电影详情 | 展示电影的详细信息，包括简介、评分、评论等 |
| 管理电影 | 管理员可以对电影进行上架、下架和编辑电影信息 |
| 收藏电影 | 用户可以收藏自己感兴趣的电影 |
| 评论模块 | 发表评论 | 用户可以对电影发表评论和评分 |
| 评论列表 | 展示电影的所有评论，支持分页和排序 |
| 评论删除 | 用户可以删除自己的评论 |
| 管理评论 | 管理员可以对用户发布的评论进行管理 |
| 情感分析模块 | 评论情感分析 | 使用BERT模型分析评论的情感倾向 |
| 情感趋势 | 展示电影评论的情感趋势变化 |
| 数据分析模块 | 用户行为分析 | 分析用户活跃度和行为模式 |
| 电影评分分析 | 分析电影评分分布和趋势 |
| 电影类型分析 | 可以查看系统中上架电影的类型 |
| 影评关键词分析 | 可以查看该部电影影评关键词情况 |
| 系统管理模块 | 用户权限管理 | 管理员可以管理用户权限和状态 |
| 数据备份 | 支持系统数据的备份和恢复 |

* + 1. 系统数据分析

在系统数据建模阶段，本文采用Pydantic库来定义实体的数据结构，通过严格的数据类型校验，简化了FastAPI框架中的API序列化过程。数据模型主要包括以下4个：

* 用户模型 (users)：\_id, user\_id, username, email, password, avatar, role, status, created\_at。
* 电影模型 (movies)：\_id, title, director, actors, genres, release\_date, rating, poster, description, created\_at。
* 评论模型 (reviews)：\_id, movie\_id, user\_id, content, rating, sentiment, created\_at。
* 喜好集合 (favorites)：\_id, user\_id, movie\_id, created\_at。

在系统数据建模中，用户、电影、评论、喜好四个模型之间存在多个一对多的关联关系。一个用户可以发表多条评论、可以收藏多部电影，因此用户与评论、喜好都是一对多关系；每部电影可以拥有多条用户评价、被多个用户收藏，所以电影与评论、喜好是一对多关系。这些关联共同构成了系统的基础数据关系网络。

* + 1. 系统功能分析

本系统中定义了两种角色，分别是普通用户和管理员用户。普通用户拥有的系统功能包括用户注册、用户登录、个人信息管理、电影信息查看、电影收藏、评论查看等，管理员用户拥有的功能还包括用户权限管理、用户行为分析、电影列表管理、评论列表管理、评论情感分析、电影评分分析等，对应的系统用例图如图3-1所示，对应系统的E-R图如3-2所示：

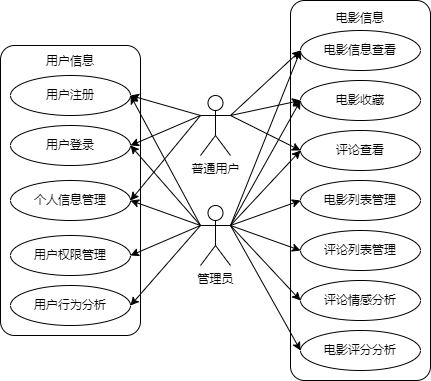


图3-1 系统用例图

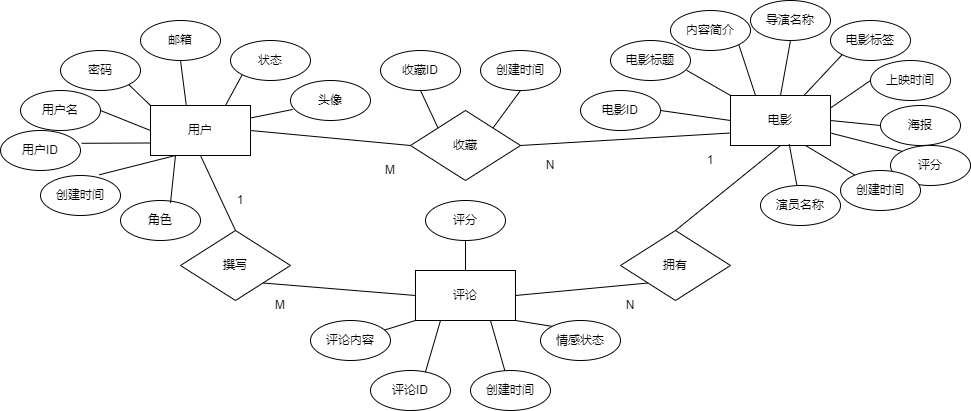


图3-2 系统E-R图

* 1. 系统概要设计
     1. 系统设计概述

本系统选用了前后端分离的架构模式。前端选用Vue.js 3框架，使用组合式API进行代码编写，组件库选用Ant Design Vue，让用户界面风格统一、专业，系统路由管理选用Vue Router，系统状态管理选用Pinia，要求设计好页面展示逻辑，功能分离。

后端服务通过FastAPI框架构建，包含API接口文档，要求设计满足前端功能的各项后端接口。数据持久化方案选用MongoDB，设计Motor库实现异步数据库访问。

* + 1. 系统数据库设计

数据库设计采用MongoDB非关系型数据库，系统中的数据集合主要包括用户集合、电影集合、评论集合、喜好集合四个。其中用户集合实现了用户基本信息和权限管理，支持用户数据的快速查询；电影集合实现了电影信息的存储，包括基本信息、评分、评论等，支持多维度查询；评论集合实现了评论数据的存储，包括内容、情感分析结果等，支持评论的快速检索；喜好集合实现了用户对电影的收藏，包括收藏时间等。在数据库结构上，系统实现了数据库索引优化、查询性能优化等调整，提高了数据访问效率。系统数据库各集合信息如下：

（1）用户数据集合

用户数据集合中，username和email设置为字符串类型，其中email使用了EmailStr类型，能够确保字符串符合电子邮件的格式规范；password字段存储的是经过bcrypt算法加密后的哈希值，而非密码的明文，能够保障数据安全；avatar为可选字符串类型Optional[str]，意为允许用户不设置头像；role和status字段包含默认值（'user', 'active'），为后续的角色权限管理和用户状态控制保证默认情况；created\_at字段使用datetime类型记录用户创建时的时间戳，便于后续的图表追踪和分析。

class User(BaseModel):  
 id: str  
 username: str  
 email: EmailStr  
 password: str  
 avatar: Optional[str] = None  
 role: str = "user"  
 status: str = "active"  
 created\_at: datetime

（2）电影数据集合

电影数据集合中，title, director, description等文本信息采用字符串类型，演员actors和类型genres为字符串列表，支持多值存储和查询；release\_date使用date类型，rating评分数据使用float类型。

class Movie(BaseModel):  
 id: str  
 title: str  
 director: str  
 actors: List[str]  
 genres: List[str]  
 release\_date: date  
 rating: float  
 poster: str  
 description: str  
 created\_at: datetime

（3）评论数据集合

评论数据集合中包含了评论内容content、用户评分rating以及情感倾向sentiment，其中 user\_id和movie\_id为外键。这些Pydantic模型定义好了程序内部的数据结构，同时也映射到了MongoDB数据库中的文档结构，能够确保数据在系统各层中的一致性与有效性。

class Review(BaseModel):  
 id: str  
 movie\_id: str  
 user\_id: str  
 content: str  
 rating: float  
 sentiment: str  
 created\_at: datetime

(4)喜好数据集合

喜好数据集合中包含了喜好ID、创建时间等属性，其中user\_id和movie\_id为外键。

class Favotite(BaseModel):  
 id:str  
 user\_id: str

movie\_id:str  
 created\_at: datetime

具体而言，集合设计如下：users集合存储用户信息，其中username和email字段被设置为唯一索引，以确保用户标识的唯一性。

movies集合存储电影元数据，考虑到用户经常按标题、简介或类型进行搜索，系统中创建了title和description字段的文本索引以支持高效的全文检索，同时为genres和actors这类数组字段建立多键索引以优化基于标签的查询。

reviews集合存储用户评论，其中movie\_id和user\_id字段作为外键关联，content字段和created\_at字段也建立了索引，用于评论的排序和筛选。

在处理数据关系时，考虑到评论与电影、用户的紧密关联以及查询需求，系统主要采用了引用（Referencing）的方式，即在reviews集合中存储movie\_id和user\_id，而不是将整个用户或电影文档嵌入评论中，这样可以避免数据冗余和单个文档过大的问题。

系统数据建模包括四个模型的关系建立，其中用户模型通过user\_id字段与评论模型和喜好模型形成一对多关系，一个用户可以发表多条评论，user\_id作为外键存在于reviews表中，也可以收藏多部电影，user\_id作为外键存在于favorites表中。同样，电影模型通过\_id字段与评论模型和喜好模型建立一对多关联，使得每部电影可以拥有多条用户评论，movie\_id作为外键存在于reviews表中，也可以被多个用户收藏，movie\_id作为外键存在于favorites表中。评论模型作为中间实体，同时关联用户和电影，完整记录了"谁(users)对哪部电影(movies)发表了什么评论"这一完整业务逻辑。这种关联结构有效支持了系统中的用户互动、内容评价和个性化收藏等核心功能。其具体关系如图3-3所示：

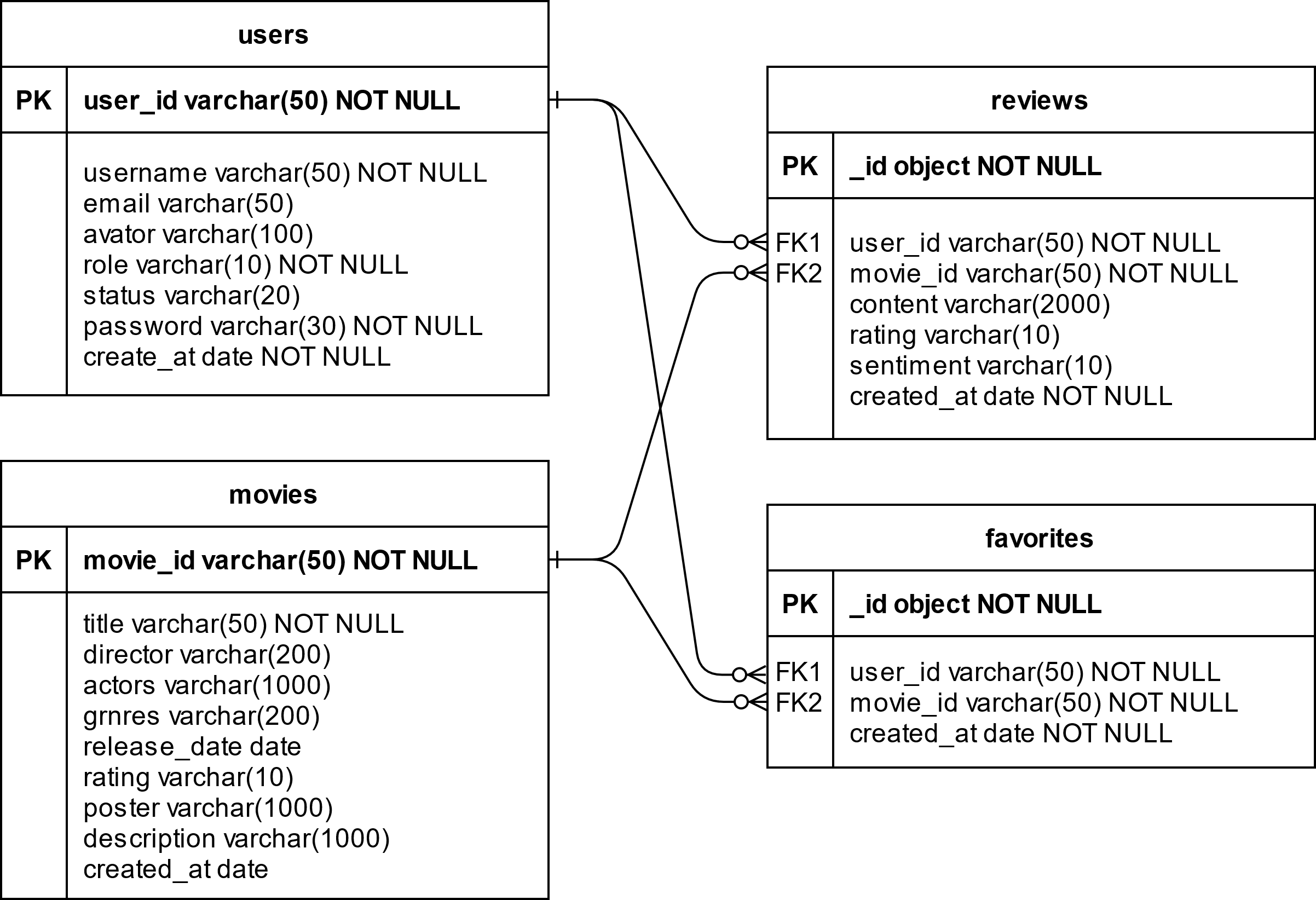


图3-3 数据关系图

* + 1. 系统用户界面设计

系统用户界面主要通过Ant Design Vue组件库构建页面，优点是可以保持页面视觉风格上的一致性和专业性。所设计的系统用户页面主要包括登录界面、电影列表界面、电影详情界面、数据分析界面四个部分：

1. 登录界面

登录页面（UserLogin.vue）主要包括用户名/密码输入框、登录按钮、注册链接等选项，提供了必要的输入引导和表单验证（如用户名、密码非空），支持回车提交，并通过按钮的加载状态（loading）和禁用（disabled）属性防止重复提交，提升交互友好性。

1. 电影列表界面

电影列表界面（MovieList.vue）主要包括电影卡片网格布局、搜索和筛选功能、分页控件、排序等选项，采用响应式的卡片网格布局（a-card, a-row, a-col），在不同屏幕尺寸下都能良好展示；此外页面集成了搜索框（MovieSearch.vue）和基于类型、评分等的筛选器（a-select, a-dropdown），支持即时反馈；通过分页控件（a-pagination）实现了数据懒加载，避免一次性加载过多数据。

1. 电影详情界面

电影详情界面（MovieDetail.vue）主要包括电影海报、基本信息展示、评分和评论、评论表单等功能，包括展示电影海报（a-image支持）、电影基本信息（a-descriptions结构化）、用户评分（a-rate）以及评论列表（ReviewList.vue）。

1. 数据分析界面

数据分析界面（UserAnalytics.vue, MovieAnalytics.vue, ReviewAnalytics.vue）主要包括情感分析图表、用户行为分析、评分分布图、评论趋势图等图表可视化组件。通过利用ECharts的图表绘制能力，构建了情感分布饼图、评分分布直方图、情感/用户活跃度趋势折线图、词云图等多种交互式图表。

* + 1. 系统API接口设计

系统接口采用FastAPI统一管理，设计遵循RESTful（Representational State Transfer）架构风格。资源的识别通过URL路径（如/api/users/me, /api/movies/{movie\_id}）实现，操作类型则由HTTP方法（GET, POST, PUT, DELETE）定义。

具体而言，系统通过POST /api/users/register用于创建新用户，GET /api/movies用于获取电影列表，GET /api/movies/{movie\_id}通过路径参数获取特定电影详情等。

请求参数的传递方式依据场景选择：①路径参数用于标识特定资源，②查询参数（如?skip=0&limit=10）用于分页、排序或筛选，③请求体则用于传输复杂数据结构（如用户注册信息、评论内容），并利用Pydantic模型自动进行数据验证和解析。

安全性方面，对于需要认证的接口，如获取用户信息/api/users/me、发表评论/api/reviews等，系统结合OAuth2密码流和JWT实现认证功能。前端在登录后获取access\_token，并在后续请求的Authorization头中携带Bearer <token>，后端通过Depends(get\_current\_user)这样的依赖项自动验证Token并获取当前用户信息，无效或缺失Token的请求将被拒绝，返回401 Unauthorized报错。

主要接口项包括：用户相关接口、电影相关接口、评论相关接口、喜好相关接口、分析相关接口等。API响应采用JSON格式，成功操作返回200 OK或201 Created状态码及相应数据，客户端错误返回4xx状态码，例如422 Unprocessable Entity，服务器内部错误返回5xx状态码。接口定义遵循OpenAPI 3.1.0规范，FastAPI能据此自动生成交互式的Swagger UI和ReDoc文档，便于在后端进行API的调试。

* 1. 本章小结

本章围绕电影评论情感分析系统的需求分析与设计展开，详细阐述了系统功能模块划分、基于MongoDB的数据建模、前后端分离架构设计、Fast API接口设计等。系统实现评论情感分析，支持用户管理、电影浏览、评论交互及数据可视化功能，采用角色权限控制机制区分普通用户与管理员的访问权限，并通过索引优化和异步数据库访问提升系统性能。

# 系统详细设计

* 1. 实现环境与工具
     1. 硬件环境

系统中用到的硬件环境如表4-1所示。

表 ‑1 硬件环境

|  |  |  |
| --- | --- | --- |
| CPU | GPU | OS |
| Intel ® Core(TM) i7-12650H 2.30GHz | NVIDIA RTX 4060 6G | Windows 10 64位操作系统 |

* + 1. 实现工具

系统在构建过程中使用到的实现工具如表4-2所示。

表 4‑2 实现工具列表

|  |  |  |
| --- | --- | --- |
| **工具名称** | **版本** | **用途** |
| WebStorm | 2024.3 | 前端开发和设计主要IDE |
| MongoDB | 8.0.6 | 数据库 |
| PyCharm | 2024.1 | 后端开发以及网页爬虫主要IDE |
| Python | 3.11.9 | 后端开发主要使用的编程语言 |
| MongoDB Compass | 1.46 | 数据库管理工具 |
| Node.js | 22.12.0 | 前端开发所需环境 |
| TypeScript | 5.8.3 | 前端开发主要使用的编程语言 |
| Docker | 28.0.1 | 容器化部署 |
| Git | 2.41 | 版本控制 |

* 1. 用户登录模块

用户管理模块是系统的核心组成部分，包含完整的用户生命周期管理。

在用户注册环节，系统采用Pydantic进行数据验证，确保用户信息的完整性和有效性，登录时采用JWT认证机制，实现无状态的用户会话管理，在存储时通过bcrypt算法对密码进行加密存储，保障用户数据安全。

个人信息管理功能支持用户头像上传、邮箱修改等操作。包含于区分普通用户和管理员权限，确保系统资源的安全访问。在用户体验方面，实现了自动登录等便捷功能，同时提供了完整的错误处理和友好的用户提示。

* + 1. 用户认证实现

系统中采用了基于OAuth2密码流和JWT的认证方案。其流程如下：用户先在前端提交用户名和密码至/api/users/token，后端接收到凭证后，从数据库中查询用户是否存在，然后使用4.2.2节中描述的verify\_password函数比对其提交的密码与数据库中存储的哈希值，验证通过后，后端将用户的唯一标识user\_id以及必要的元数据作为Payload，使用预定义的密钥（SECRET\_KEY）和HS256算法生成JWT access\_token。此Token返回给前端后,将其存储在浏览器的localStorage、sessionStorage中。使用JWT进行身份认证的代码实现如下，包括在请求头中携带token，验证、提取token信息等操作，如代码4-1所示。

代码4-1

# 用户认证中间件

async def get\_current\_user(token: str = Depends(oauth2\_scheme)):

credentials\_exception = HTTPException(

status\_code=status.HTTP\_401\_UNAUTHORIZED,

detail="Could not validate credentials",

headers={"WWW-Authenticate": "Bearer"},

)

try:

payload = jwt.decode(token, SECRET\_KEY, algorithms=[ALGORITHM])

username: str = payload.get("sub")

if username is None:

raise credentials\_exception

except JWTError:

raise credentials\_exception

user = await get\_user(username)

if user is None:

raise credentials\_exception

return user

对于需要认证的API请求，前端在HTTP请求的Authorization头中以Bearer <token>的形式附带此Token。后端FastAPI应用中，定义了一个依赖项函数get\_current\_user，通过OAuth2PasswordBearer(tokenUrl="token")来自动从请求头中提取Bearer Token，解码并验证Token的有效性，即可验证签名是否有效。从Token的Payload中提取用户标识后，据此查询数据库以获取完整的用户对象用于验证签名。如果Token无效或用户不存在，则抛出HTTPException 401 Unauthorized的报错，中断请求处理，否则将查询到的用户对象返回。

* + 1. 密码加密实现

密码加密使用bcrypt算法进行密码加密，具体实现借助了Python的passlib库中的各种密码哈希算法。在系统中我首先配置了一个CryptContext实例，指定使用bcrypt方案为HAS256算法。当用户注册或修改密码时，调用pwd\_context.hash(plain\_password)函数，为密码生成一个随机的盐（salt），并将盐与密码结合后进行多次哈希运算，最终生成一个包含算法信息、计算成本、盐和哈希值本身的字符串，存储在数据库的password字段中。

当用户登录进行密码验证时，调用pwd\_context.verify(plain\_password, hashed\_password\_from\_db)函数进行解码运算。具体核心代码4-2所示：

代码4-2

# 密码加密

def get\_password\_hash(password: str) -> str:

return pwd\_context.hash(password)

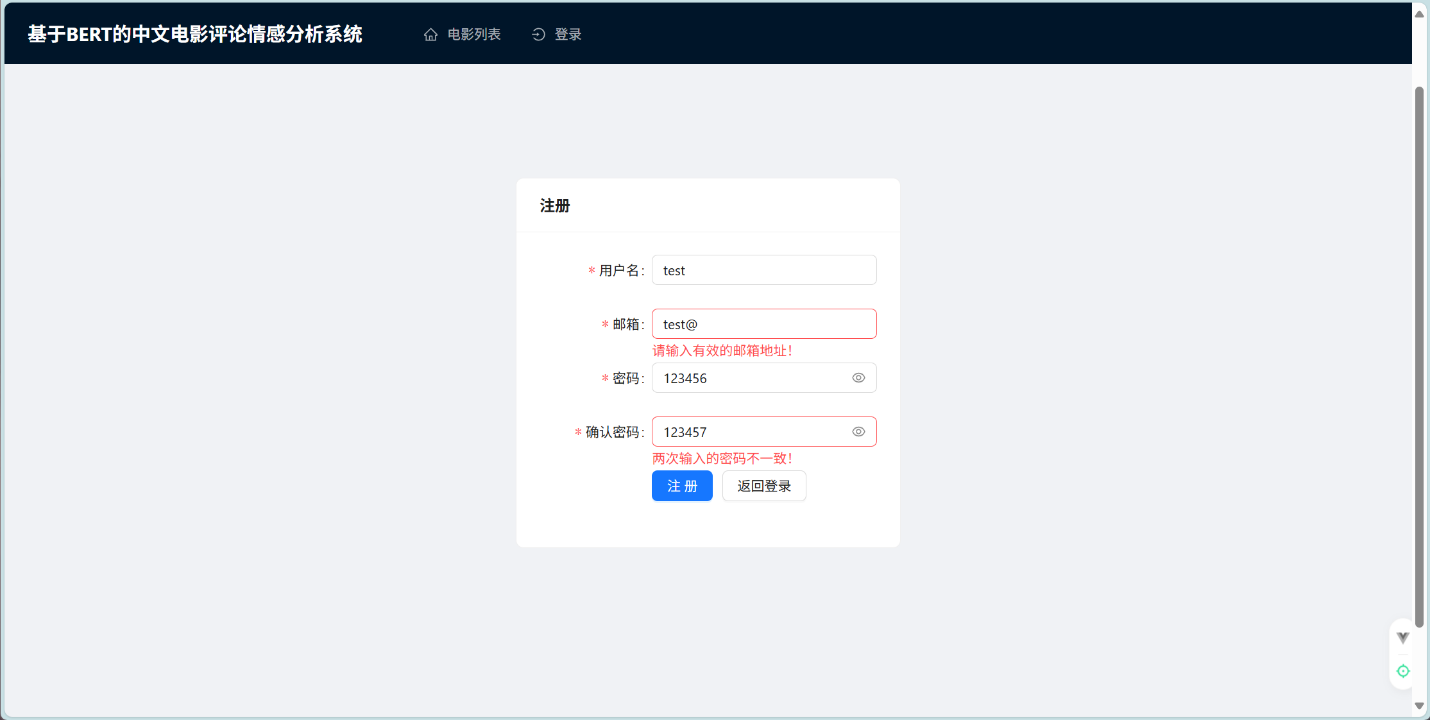
# 密码验证

def verify\_password(plain\_password: str, hashed\_password: str) -> bool:

return pwd\_context.verify(plain\_password, hashed\_password)

passlib会自动从存储的哈希字符串中提取出盐和哈希参数，用相同的盐和参数处理用户输入的明文密码，比较生成的哈希值与数据库中存储的哈希值是否一致。这种方式确保了即使数据库泄露，攻击者也无法直接获取用户明文密码，且由于每个密码都有独立的盐，也不能通过相同哈希来推断密码。

未注册的用户可以点击注册按钮进行注册，如图4-1所示，用户注册需要输入自己的用户名，注册邮箱，自己设置的密码并再次确认自己的密码，如果注册的邮箱不存在则会报错，邮箱地址无效；如果用户设置的密码，两次不一样则会报错两次输入的密码不一致，从而无法完成注册功能。

 图4-1 用户注册界面

已注册的用户可以选择输入密码进行登陆，需要输入正确的用户名和密码，如图4-2所示。如果用户名或者密码不正确，则会进行提醒，用户名或密码错误，用户将无法进行登陆。

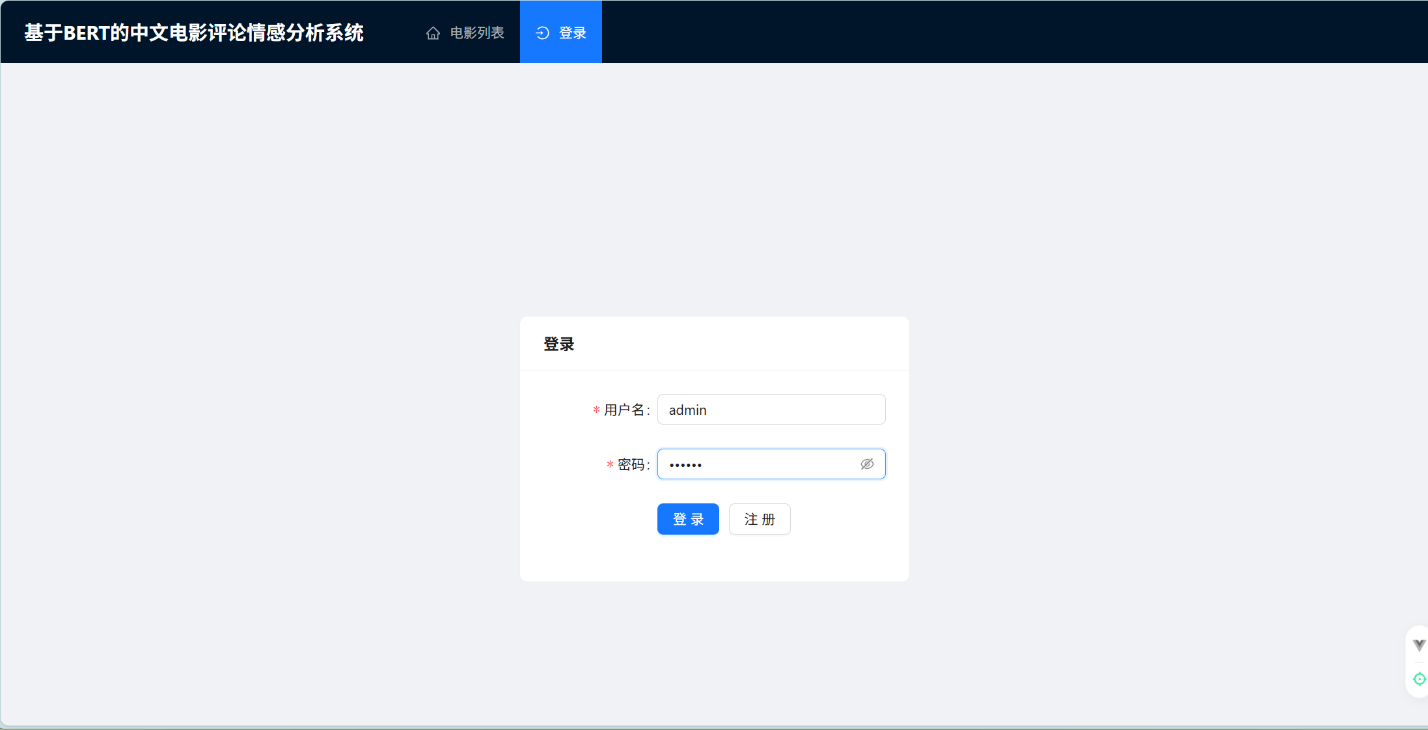


图4-2 用户登陆界面

* 1. 电影展示模块

电影管理模块实现了电影数据的全生命周期管理。首先通过异步爬虫技术从爱奇艺等平台获取电影数据，包括电影基本信息、剧情简介等，随后在数据清洗环节对爬取的数据进行标准化处理，确保数据质量，这里使用到了正则表达式匹配和自然语言处理技术等。

在电影展示中，模块采用响应式网格布局，支持按类型、评分等多维度筛选，电影海报的懒加载和渐进式加载，优化了页面性能。

* + 1. 电影数据集爬取

数据爬取流程如图 4-3所示，对应的实现代码见代码 4-3。以爬取爱奇艺平台的影评数据为例，具体步骤如下：

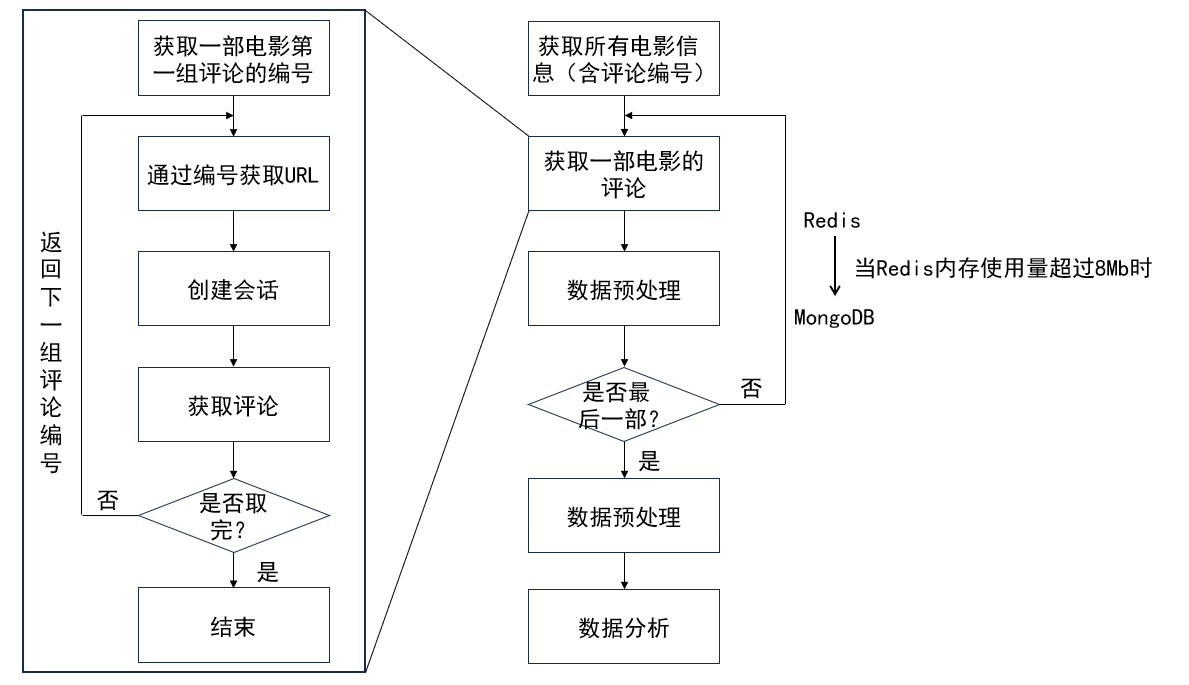
* 系统首先根据地区与语种（如华语、英语等）分类，调用平台接口获取当前热门电影的基本信息列表。
* 每部电影在平台上通常对应一个唯一的编号。通过该编号可以准确定位到该影片对应的评论接口。
* 分页获取评论数据：由于单次请求返回的评论数量有限，为获取足够的数据量，需要多次调用评论接口。通过不断更新接口中的 lastId 参数（该参数名称依平台而异，爱奇艺平台中为 lastId），即可实现评论数据的分页获取。

图 4-3 爬虫流程

代码4-3

def get\_film\_msg(url\_data\_msg\_interface, headers):

response = requests.get(url\_data\_msg\_interface, headers=headers)

data = json.loads(response.text)

films\_msg = []

for x in data['data']:

# 此处的x包含了一部影片包含的所有的基本信息

display\_name = x['display\_name'] if 'display\_name' in x else ''

display\_name = display\_name.replace('（普通话）', '')

display\_name = display\_name.replace('（独家修复版）', '')

display\_name = display\_name.replace('《', '')

display\_name = display\_name.replace('》', '-')

album\_id = x['album\_id'] if 'album\_id' in x else ''

description = x['description'] if 'description' in x else ''

tag\_pcw = x['tag\_pcw'] if 'tag\_pcw' in x else ''

url\_film = x['url'] if 'url' in x else ''

image\_url\_normal = x['image\_url\_normal'] if 'image\_url\_normal' in x else ''

film\_msg = {'display\_name': display\_name, 'album\_id': album\_id, 'description': description, 'tag\_pcw': tag\_pcw,

'url\_film': url\_film, 'image\_url\_normal': image\_url\_normal}

films\_msg.append(film\_msg)

print(film\_msg)

return films\_msg

步骤2和步骤3的代码实现如代码4-4所示，其中lastId可以在当前接口请求到的数据中找到，并通过lastId可以找到下一组评论的接口。此处用异常的方式结束爬取，当全部评论爬取完后，会自动中断while循环。

另外，在代码中创建了requests的session对象，该对象在while循环外持久创建，并以参数的形式传递给getMovieinfo函数（具体实现见下述代码）。这样做可以防止过多建立连接，减少请求次数，并加快爬取速度。

代码4-4

def getMovieinfo(url, session, headers):

try:

response = session.get(url, headers=headers)

if response.status\_code == 200:

return json.loads(response.text)

else:

return None

except Exception as error:

print(error)

return None

最后将数据封装在字典内，该字典可作为MongoDB的一个文档直接存储。

爬虫使用aiohttp进行异步HTTP请求，设置User-Agent模拟浏览器，实现了分页爬取、异常处理和重试机制、数据去重和验证等功能。

* + 1. 数据清洗

从网络爬取或多源整合的数据往往存在格式不一、信息缺失、包含无关字符等问题，需要在数据获取后进行清洗。在数据获取阶段，我们取得了个数如图4-4所示的一组数据，在本阶段，会对这些数据进行预处理。

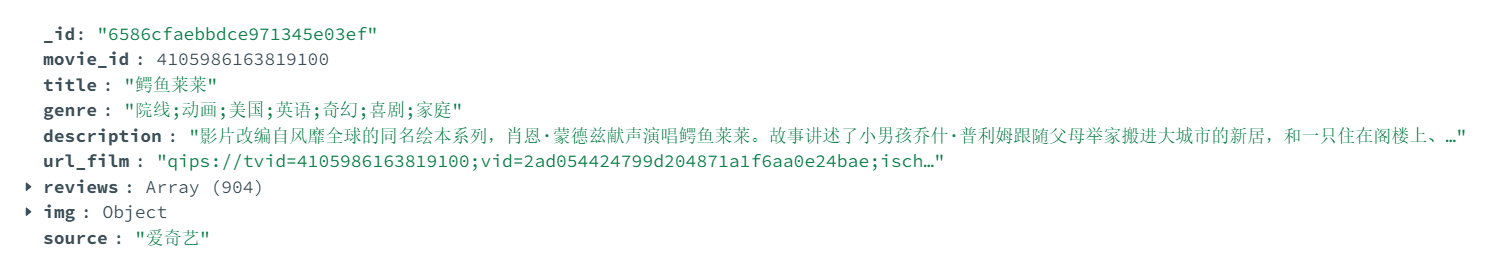


图 4‑4 数据预处理展示

本系统的数据清洗流程主要针对电影元数据，采取了一系列自动化处理规则。例如，对电影标题title、简介description等文本字段，进行了首尾空白字符去除、特殊HTML实体转义（如&amp;转为&）、非标准字符过滤等操作。对于演员actors和类型genres这类通常由特定分隔符连接的字符串，先要进行标准化拆分，转换为字符串列表，再对列表中的每个元素进行独立清洗，从而去除空格等无效字符。

对于上映日期release\_date字段，需要处理多种可能的日期格式（如'YYYY-MM-DD', 'YYYY年MM月DD日'），并统一转换为标准的ISO日期格式。对于评分rating字段，需要处理其中的非数字字符，并转换为浮点数类型。

数据清洗脚本被设计为幂等的，即对同一份原始数据多次运行清洗脚本，结果是保持一致一致的。对于无法通过自动化规则处理的异常数据或缺失值，会进行异常值记录，通过人工干预填充数据，具体代码如代码4-5所示：

代码4-5

def clear\_special\_char(content):

"""

正则处理特殊字符

参数 content:原文本

return: 清除后的文本

"""

s = re.sub(r"</?(.+?)>|&nbsp;|\t|\r", "", content)

s = re.sub(r"\n", " ", s)

s = re.sub(r"\\*", "\\\*", s)

s = re.sub('[^\u4e00-\u9fa5^a-z^A-Z^0-9]', "", s)

# 去除不可见字符

s = re.sub(

"[\001\002\003\004\005\006\007\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a]+",

"", s)

s = re.sub('[a-zA-Z]', '', s)

s = re.sub('^\d+(\.\d+)?$', "", s)

return s

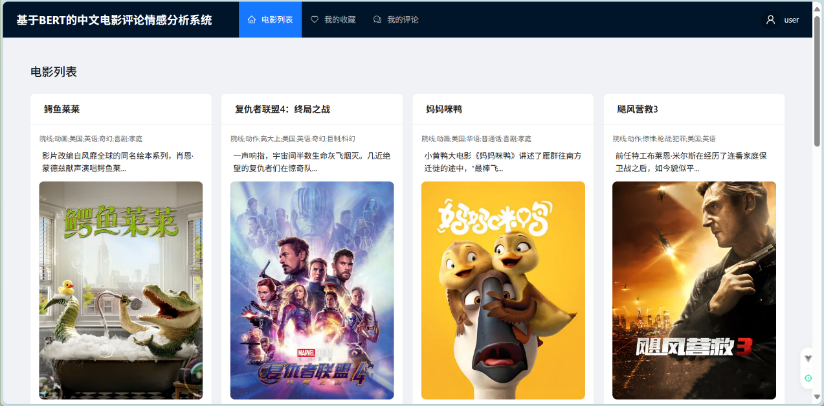
用户在登陆系统之后就可以查看系统中有的电影，并在点击电影海报后，可以进入该部电影详细信息界面，如图4-5所示：

图4-5 电影信息查看功能

用户可以收藏自己感兴趣的电影，从而帮助用户进一步缩小选取电影范围，用户点击收藏之后，根据登陆用户的id，在favorite数据表中，添加对应的movie\_id，从而完成用户收藏，用户在点击“我的收藏”之后就可以查看自己收藏的电影，如图4-6所示：

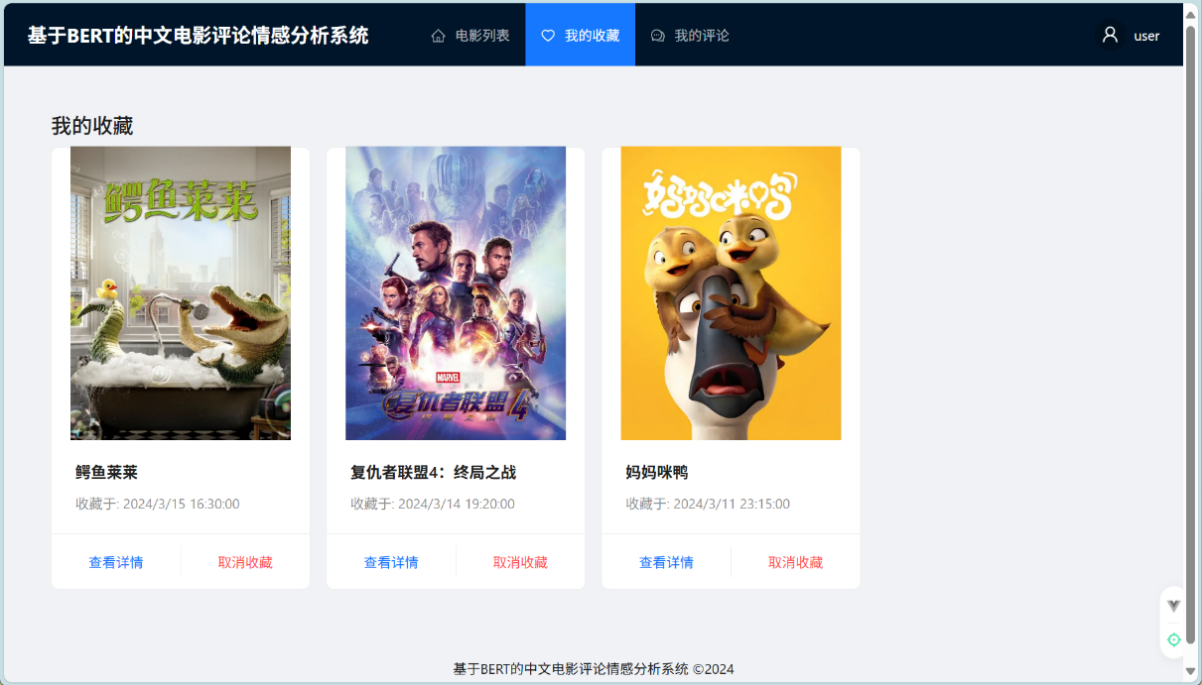


图4-6 收藏电影功能

管理员通过登陆管理员账户就可以进入管理系统，在本系统当中，管理员通过点击电影管理功能，可以查看系统中已发布的电影，包括电影的ID，电影名称，电影类型以及电影的简介，通过添加电影从而实现对电影的上架，可以点击删除从而实现对电影的下架，点击编辑电影可以实现对电影名称，类型和简介的修改，用户可以通过电影名称快速定位到自己感兴趣的电影，具体功能如图4-7和图4-8所示。

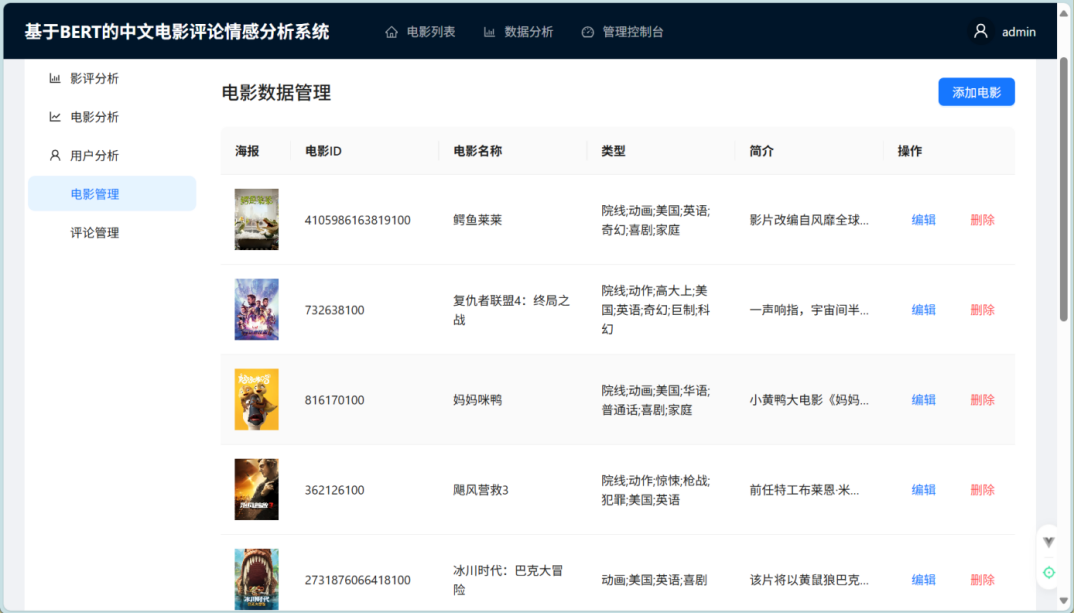


图4-7 管理员管理电影功能



图4-8 管理员管理电影功能

* 1. 电影评论分析模块

评论管理模块实现了用户评论的完整生命周期管理。系统采用BERT模型对用户评论进行初步处理，随后通过监督学习，分类器进行评论情感分析分类，模型将评论情感分为积极、中性和消极三类，通过计算情感置信度可视化展示。评论展示功能主要采用分页加载和虚拟滚动技术，优化了性能。

* + 1. 电影评论数据爬取

电影评论数据的获取同样依赖于异步网络爬虫。与电影元数据爬取类似，评论爬取也需模拟浏览器等行为，在处理分页、解析HTML结构后，提取到每部电影的评论内容、用户名等信息。爬取原理同4.3.1节的部分。

爬取评论数据后，需要先对评论数据进行中文分词。系统中使用的分词库为在中文分词领域广泛使用的jieba库，对于每条评论text，调用jieba.cut(text)方法进行分词。jieba支持多种分词模式，系统中选用了精确模式以获得最准确的切分结果。

考虑到电影评论中可能包含领域特定的术语，通过jieba.load\_userdict('user\_dict.txt')加载自定义词典。分词后，通过停用词过滤，将那些在文本中频繁出现但对语义贡献不大的词语，例如"的"、"了"、"是"、标点符号等处理掉，其中包括维护了一个停用词表（stopwords.txt），可以在分词结果中剔除这些词语。

* + 1. 数据格式转换

为了将处理后的文本输入到BERT模型中，必须将其转换为模型能够理解的特定数值格式。这一步主要由Hugging Face Transformers库提供的BertTokenizer.from\_pretrained ('bert-base-chinese')完成。首先加载预训练tokenizer，通过文本编码、填充和截断等方式转换为PyTorch张量。

同时在预处理时将数据格式统一。例如，某些电影的类别用字符串表示，这里将其改为列表。效果如图4-9所示：

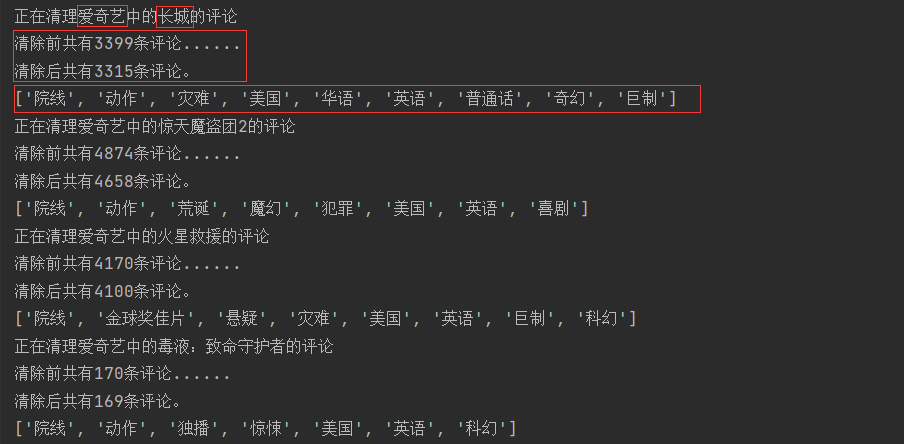


图4-9 数据格式处理示例

* + 1. 评论数据可视化实现

评论数据展示与电影信息展示类似，通过使用v-for指令遍历电影评论数据列表，将影评的信息在循环体内渲染a-card，数据将在卡片内详细展示，如图4-10所示。



图 4-10 评论数据展示页面

除此之外，通过统计评论长度，结合Echart饼图，能够可视化展示评论的分布情况，如图4-11所示。



图 4‑11 评论数据分析实现

* 1. 情感分析模块
     1. 情感分析的模型框架

情感分析模块部分主要采用基于BERT预训练模型进行中文文本情感分析。系统对BERT模型进行了针对性的微调，通过使用电影评论数据集进行训练，在文本预处理、分词、特征提取和分类预测等步骤后，实现了对评论情感的识别，通过Echart可视化展示，包括情感分布饼图、情感趋势折线图等，帮助用户直观理解评论情感走向。在性能优化方面，实现了模型推理的批处理机制和缓存策略，提高了系统的响应速度。模型流程图如4-12所示。

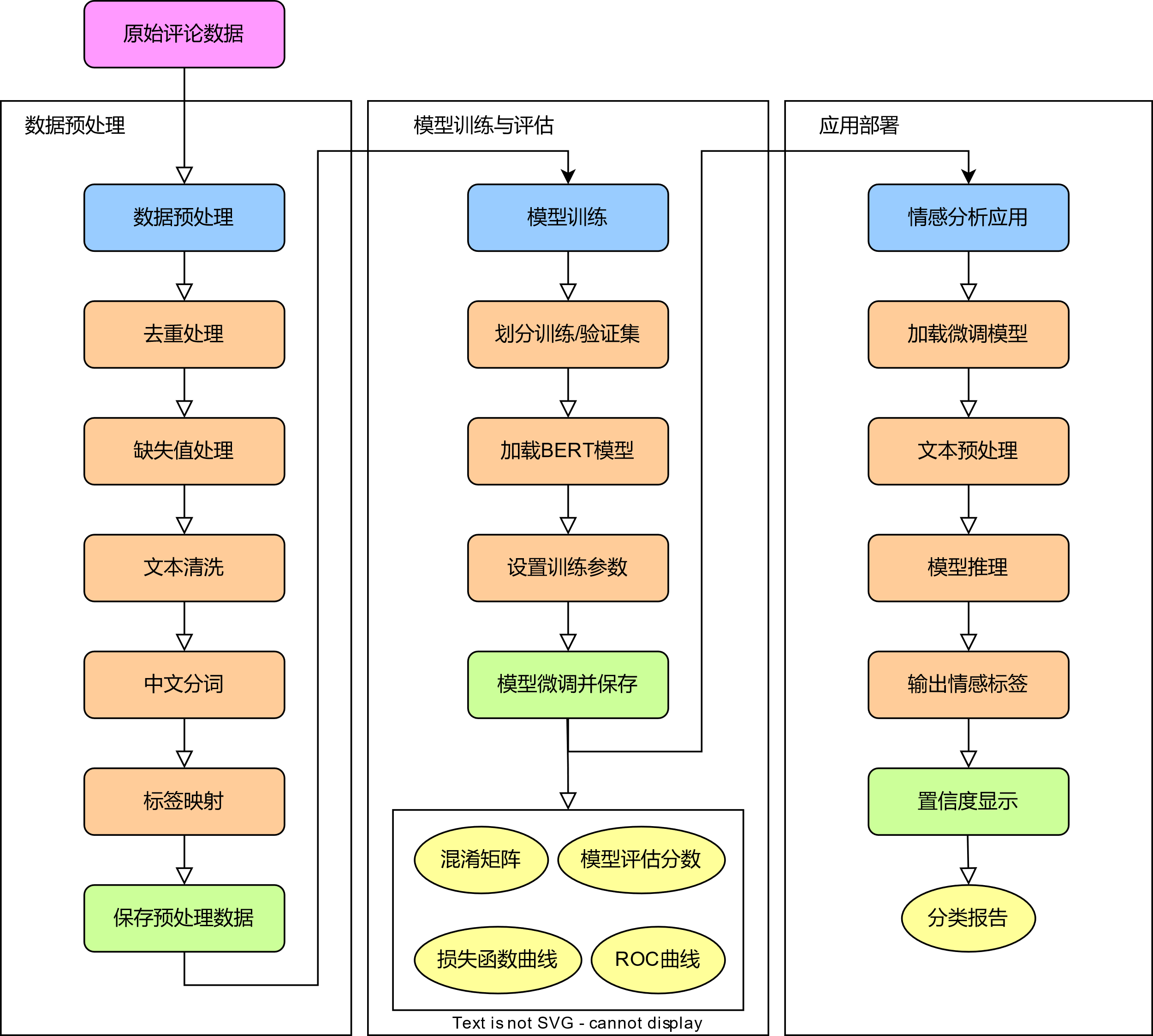


图 4‑12 情感分析模块流程图

本系统的情感分析模块基于 BERT 深度学习架构构建，采用了中文预训练模型 **bert-base-chinese**。该模型在大规模中文语料上训练完成，具备良好的语法理解与语义建模能力，在保持较高性能的同时，对计算资源的依赖适中，适合在现有硬件环境下进行微调。经过微调后，模型被用于电影评论的情感分类任务。

情感分析过程主要包括三个阶段：文本编码、特征提取与分类判定。首先将原始评论文本输入 BERT 模型进行上下文编码，提取语义特征；随后通过分类器对这些特征进行处理，输出情感倾向的概率分布；最后根据概率分布进行分类判定。

系统采用BERT-base-Chinese预训练模型作为基础架构，其数学形式可用公式4-1表示：

(4.1)

1、输入表示

嵌入层，将输入序列转化为向量表示，如公式4-2所示：

(4.2)

公式4.2中 TokenEmbed 为词嵌入矩阵，PositionEmbed 为位置编码，SegmentEmbed 为句子类型编码（本系统单句场景下恒为0）。

2、编码器

由 L=12 个相同层堆叠而成，每层编码器包含多头自注意力机制和前馈神经网络，如公式4-3和公式4-4所示：

(4.3)

(4.4)

**（1）多头注意力机制（）**

表示多头注意力，Q为查询矩阵，表示需要计算注意力的目标序列，K为键矩阵，表示被检索的上下文序列，V是值矩阵，表示实际需要加权聚合的信息序列，是第i个注意力头的输出结果，那么在BERT-base-Chinese模型中，此时公式4.3中的h即为12，是输出权重矩阵，用于将拼接后的多头输出投影到最终维度，最后通过（.）将多个头的输出沿最后一维拼接。

公式4.4则表示了每个头的计算过程， 其中和分别表示查询权重矩阵和键权重矩阵，表示值权重矩阵，表示缩放因子，防止点积结果过大导致Softmax梯度消失，最后（.）对注意力分数归一化，生成权重分布。

**（2）前馈神经网络（FNN）**

每层注意力后接了一个两层全连接网络，如公式4.5所示：

（4.5）

其中，表示多头注意力后的输出，和分别表示前馈神经网络中第一层和第二层的权重矩阵，和分别表示第一层和第二层中的偏置，最后通过激活函数GELU（.）得到最后的输出。

**（3）残差连接与层归一化**

每子层（注意力/FFN）后应用公式4.6：

（4.6）

其中，公式4.6中LayerNorm（.）表示层归一化，缓解梯度消失或者爆炸问题，Sublayer（.）则是指多头注意力子层，即多头注意力层和前馈神经网络层。

通过编码器，可以得到最终层的输出[CLS]向量作为句子整体表示，用于分类任务；其余Token向量可用于标注或问答任务，本文只涉及分类任务。

3、分类器

在研究首先对预训练模型进行参数解冻，在其输出层之上添加了一个简单的分类模块。该模块由一个全连接层和 Softmax 激活函数构成，用于输出各类别的情感倾向概率。通过预处理后的电影评论数据集对整个模型进行微调训练，使模型能够进一步适应电影评论这一特定领域的语言风格和情感表达方式，从而提升情感分类的准确性与鲁棒性。在BERT编码输出的[CLS]标记向量基础上，构建分类网络，如公式4-7所示：

(4.7)

其中，为分类权重矩阵，为偏置项，输出维度3对应积极、中性、消极三类情感。Softmax函数确保概率归一化：

(4.8)

4、损失函数

损失函数的构建采用了适用于多分类任务的标准交叉熵损失函数（CrossEntropyLoss），通过计算模型预测概率分布与真实标签分布之间的差异来指导模型参数的优化方向，有效惩罚错误预测，在计算交叉熵损失、标签处理、输出转换、损失计算、梯度更新等操作后，结合模型优化部分的策略，即可找到最佳速度下降的损失函数。

本文采用带标签平滑的交叉熵损失函数优化模型，如公式4-8所示：

(4.8)

其中为平滑系数，为批次大小。

综上，给定评论文本T，其情感分析流程为：

1. 文本预处理：

2. 向量编码：

3. 特征提取：

4. 分类决策：

最终输出即为三分类概率分布。系统设定当时判定为积极情感，时判定为消极情感，其余情况归为中性。

* + 1. 模型的训练

（1）数据集简介

数据集来自从爱奇艺爬取的1018部电影数据，其中包含4324956条影评数据等。在数据存储和预处理结束后，通过聚合查询分析可得：从爱奇艺获取3272529评论，超过2000条评论的电影有683个，占电影总数的33.8%。数据的格式可参照图4-13，以JSON数据的格式存储。其中，电影的评论信息是一个列表，在进行了数据变换后，评论也以列表的形式存储。图片在经过转换过后，也可以以文本的形式存储，其中type是图片编码前的格式。



图 ‑13 数据集结构

（2）超参数设置

在超参数的选择上，本文选定训练轮数为3（num\_train\_epochs=3），避免过高或者过低导致模型过拟合或者欠拟合；设备训练批次大小设定为16（per\_device\_train\_batch\_size=16）、评估批次大小设定为64（per\_device\_eval\_batch\_size=64），以兼顾GPU显存限制和梯度估计的稳定性；预热学习率设定为500（warmup\_steps=500）、权重衰减设为0.01（weight\_decay=0.01），帮助模型更快收敛并提高泛化能力。

（3）衰减策略调整

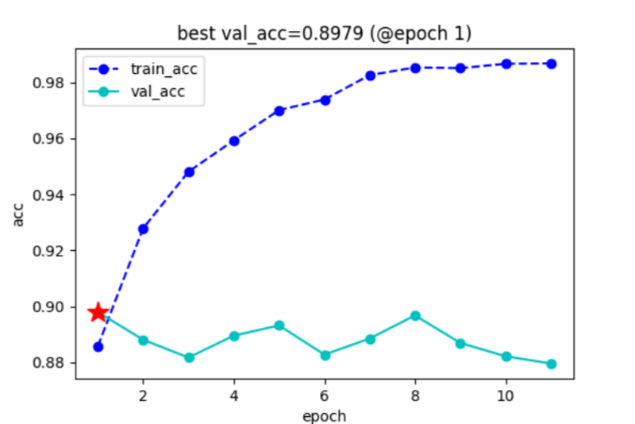
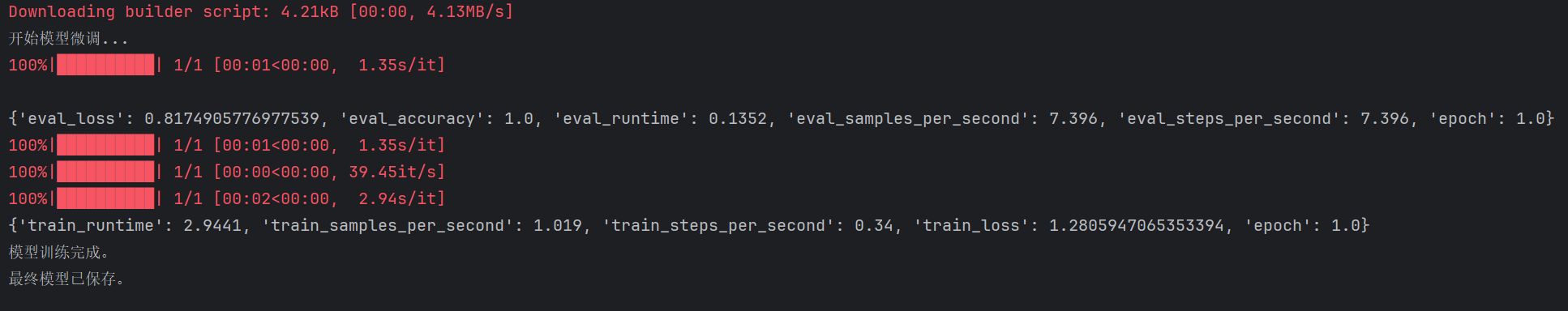
最终经过衰减策略，得到衰减值eval\_loss为1.4369511604309082, 平均每秒下降eval\_samples\_per\_second为6.803,样本损失值train\_loss大小为0.8089718222618103,epoch得到1.0，训练效果如图4-14所示：

图 ‑14 训练效果展示

训练过程使用AdamW优化器进行端到端微调，学习率设置为,权重衰减系数 0.01。训练时采用动态掩码策略，每个epoch随机遮盖15%的输入token进行MLM辅助训练。批处理大小设为32，在RTX 4060 GPU上完成5000步训练（约3小时）。

（4）模型优化

在模型优化方面，除了上述学习率调整和权重衰减策略，本文还引入了早停（Early Stopping）机制，通过监控模型在独立验证集上的F1-score，在指标不再提升时提前终止训练，可以避免模型在训练集上过拟合。同时，模型选定批量大小为16，避免过高或过低，平衡了训练速度、内存占用和模型的泛化性能。未来工作中，还可以考虑模型集成等策略，通过结合多个独立训练的模型预测结果，进一步提升整体情感分析的稳定性和准确性，微调结果如图4-15所示：

图4-15 模型微调结果

（5）模型评估

混淆矩阵评估包括模型评估模式、预测结果收集、真实标签收集等，如代码4-6所示，下图为200个样本的三分类混淆矩阵结果，如图4-16所示。

代码4-6

def evaluate\_model(model, test\_loader):

model.eval()

all\_preds = []

all\_labels = []

with torch.no\_grad():

for batch in test\_loader:

outputs = model(\*\*batch)

preds = torch.argmax(outputs.logits, dim=-1)

all\_preds.extend(preds.cpu().numpy())

all\_labels.extend(batch['labels'].cpu().numpy())

cm = confusion\_matrix(all\_labels, all\_preds)

return cm

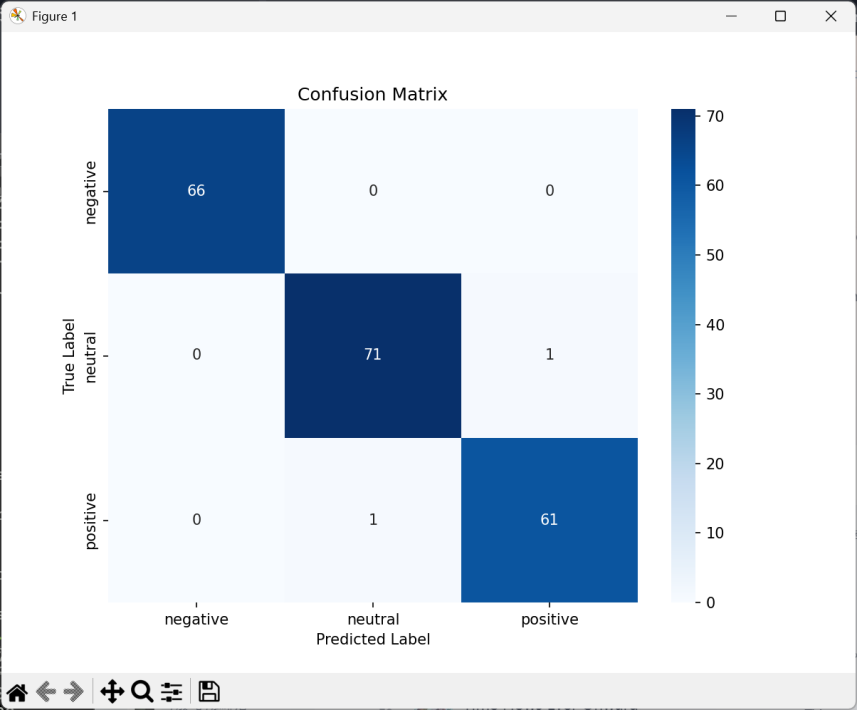


图 ‑16 混淆矩阵展示

交叉验证部分主要包括了数据集划分、模型训练、性能评估、结果统计、方差分析等步骤，防止模型训练过拟合，提高模型的泛化能力，如代码4-7所示：

代码4-7

def cross\_validate(model, dataset, n\_splits=5):

kf = KFold(n\_splits=n\_splits, shuffle=True)

scores = []

for train\_idx, val\_idx in kf.split(dataset):

train\_data = dataset[train\_idx]

val\_data = dataset[val\_idx]

# 训练模型

trainer = Trainer(

model=model,

args=training\_args,

train\_dataset=train\_data,

eval\_dataset=val\_data

)

trainer.train()

# 评估模型

eval\_results = trainer.evaluate()

scores.append(eval\_results['eval\_accuracy'])

return np.mean(scores), np.std(scores)

在本实验中，使用到的评估指标包括准确率 (Accuracy)，精确率 (Precision)，召回率 (Recall)，F1分数，ROC曲线等。

ROC 曲线展示了在不同的分类阈值下，模型的真阳性率（True Positive Rate, TPR，即Recall）相对于假阳性率（False Positive Rate, FPR）的变化情况。X 轴是 FPR (FP / (FP + TN))，Y 轴是 TPR (TP / (TP + FN))。曲线越靠近左上角，表示模型性能越好，图中对角虚线代表随机猜测的性能。

AUC (Area Under the Curve) 是 ROC 曲线下的面积，是衡量模型整体区分能力的指标，值越接近 1 越好。对于多分类问题，会绘制每个类别的 ROC 曲线（One-vs-Rest），以及宏平均（Macro-average）或微平均（Micro-average）的 ROC 曲线。其中微平均考虑了所有样本的预测结果，结果如图4-17所示：

图4-17 模型ROC曲线展示

模型的精确率、召回率、F1分数如图4-18所示：

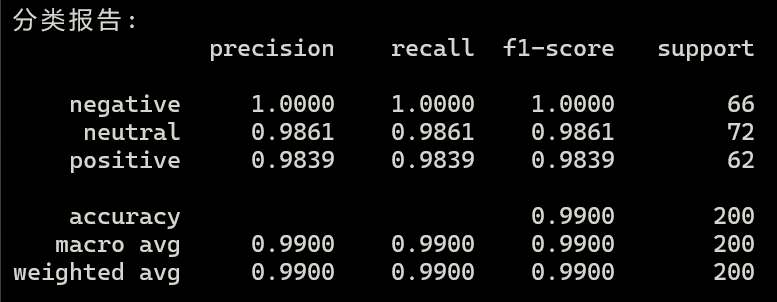


图 4‑18 模型评价指标分数展示

（6）情感分析综合推理效果

单纯依赖模型的概率输出可能不够全面，在情感内涵的理解上可能有欠缺。基于前文的研究，本文提出了一种综合分析策略，该策略考虑了模型输出的情感得分和置信度，包括设置合理的判别阈值、结合评论的具体上下文信息进行判断等。例如，识别和处理文本中的反讽、隐喻等复杂语言现象，引入了用户历史评论行为和电影本身的元数据作为辅助信息，通过Softmax输出概率来综合衡量。具体而言，结合以下因素进行综合分析：

1）情感得分：模型输出、置信度、阈值判断

2）上下文信息：评论内容、用户历史、电影信息

3）时间趋势：评论时间、情感变化、异常检测

情感分析综合推理效果如图4-19所示：

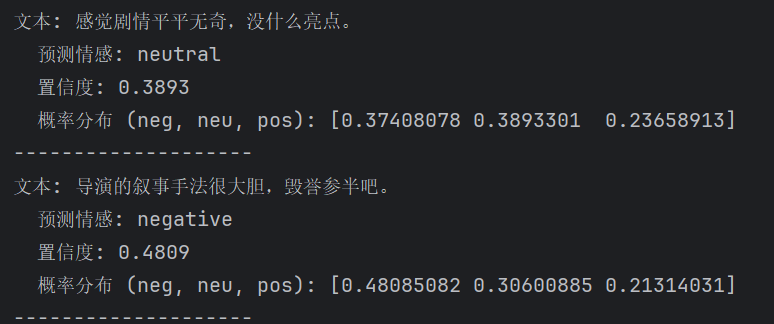


图 4‑19 情感分析综合推理结果

通过置信度等参数综合评判情感得分，以“感觉剧情平平无奇，没什么亮点”评论为例，在概率分布上，中性占比为38%更占优势，但是评论置信度一般大小，仅有38.9%，最终综合评判为中性评论。

为了直观展示用户评论中高频出现的关键词，系统对所有评论文本进行了分词处理、去除停用词，统计词频，生成了词云图。如图4-20所示，词云图以关键词出现频率为基础，通过字体大小反映词语的重要程度，能够帮助分析用户关注的核心内容与评价重点。

图 4‑20 词云图展示

* + 1. 情感分析结果可视化

在系统中，情感分析主要是利用上述基于BERT微调以及监督学习训练后的模型。当需要分析一条新的电影评论时，首先执行4.4.2节所述的数据格式转换，得到模型所需的输入张量，随后将这些张量传递给加载好的、处于评估模式下的微调模型，模型接收输入后，通过其内部的多层Transformer编码器即可完成计算，最终在分类层中可以输出每个情感类别的原始得分。为了获得概率分布，通常对Logits应用Softmax函数，概率最高的那个类别，即为是模型预测的情感倾向。

例如，一次Softmax的输出为[0.1, 0.2, 0.7]，则预测结果为0.7对应的"积极"。除了预测的类别标签，还可以通过获取对应的概率置信度，结合每个类别的具体概率完成得分统计，这些信息一同返回给了前端，用于更细致的展示或分析。

上述整个推理过程在torch.no\_grad()的上下文管理器中执行。对于需要批量处理大量评论的场景，可以将多条评论的输入打包成一在批次进行推理，以充分利用硬件并行计算能力，提高整体吞吐量。

（1）电影影评情感分布情况

根据上述逻辑，结合React Echart的饼图可以可视化展示任意一部电影的评论情感分布情况，例如《鳄鱼莱莱》电影的评论情感分布如图4-21所示：



图 4‑21 情感分析可视化

（2）电影影评随时间发布趋势

此外，结合评论的时间还可以对评论的情感趋势变化进行分析。系统中，情感趋势分析模块通过数据聚合和时间序列构建，能够分析电影评论情感随时间演变的动态变化。具体而言，在确定好时间聚合的粒度后，例如确定按天分布，通过MongoDB的聚合管道进行查询，可将评论数据按选定的时间窗口进行分组，在每个时间窗口内，进一步根据评论的情感标签列出sentiment字段，并由BERT模型预测的结果进行二次分组，并统计每个情感类别（积极、中性、消极）的评论数量。

聚合后的结果是一个时间序列数据结构，例如一个列表，其中每个元素代表一个时间点，并包含该时间点对应的积极、中性、消极评论数。例如：[{'date': '2024-03-20', 'positive': 150, 'neutral': 80, 'negative': 30}, {'date': '2024-03-21', ...}]。这个结构化的时间序列数据可以直接传递给前端的图表库用于绘制多条折线图，每条折线代表一种情感倾向的数量变化，X轴为时间，Y轴为评论数量。通过观察这些曲线的起伏、交叉和相对位置，可以分析口碑发酵过程、特定事件对情感的影响或整体舆论走向。例如《鳄鱼莱莱》电影的评论随时间变化效果如图4-22所示：



图 4‑22 情感分布趋势可视化

（3）电影影评高频词汇

除了情感分布情况，通过BERT预处理后的评论可以通过词汇分布展示词云图，能够直观展示评论文本中高频关键词的可视化技术。其生成过程主要涉及文本的预处理、词频统计和图像绘制等步骤。针对特定电影的所有评论（或者某个时间段内的评论），将其内容合并成一个大的文本字符串。接着，对这个合并后的文本进行4.3.2节所述的分词处理和停用词过滤，得到一个词语列表。然后使用Python的collections.Counter类频率即可得到一个词语到其频率的映射（字典）。最后通过利用wordcloud库来生成词云图像，初始化WordCloud对象时，需要配置多个参数，其中最关键的是font\_path，必须指定一个包含所需中文字符的字体文件路径（如simhei.ttf, msyh.ttf等），否则中文将显示为乱码方框。例如《鳄鱼莱莱》电影的评论高频词汇如图4-23所示：



图 4‑23 评论词云图可视化

* 1. 用户访问控制实现
     1. 管理员访问控制实现

管理员页面使用Ant Design布局，采用响应式设计结构，包括侧边栏导航、图标集成、路由视图等。页面包括用户管理，用户活跃度分析，电影数据编辑等。

使用管理员账号登陆进系统，并进入管理员控制台即可进行用户管理相关工作，包括查看已注册的用户名，用户名注册时所用邮箱，账户权限和删除相关账户的功能，如图4-24所示：



图 4‑24 管理员系统界面

* + 1. 普通用户访问控制实现

普通用户页面实现了表单布局、数据绑定、文件上传、表单验证、响应式设计等功能，页面包括个人信息，账号设置，我的收藏等信息。以UserProfile.vue为例，该组件的核心功能是展示和允许用户修改部分个人信息。组件在加载时，会调用后端API（如GET /api/users/me）获取当前登录用户的详细信息（用户名、邮箱、头像URL等），并将这些数据显示在表单 (a-form) 中。用户名通常是只读的。

头像部分通过使用a-upload组件允许用户选择本地图片文件，在文件选择后，前端首先会进行文件大小和类型的校验，然后通过调用后端的头像上传接口，将图片上传至服务器。服务器保存图片后，更新用户数据库中的avatar字段，并返回新的头像URL或其他成功信息。前端在收到成功响应后，更新界面上显示的头像。

普通用户登陆进系统后，会进入普通用户系统界面。如图4-25所示，点击右上方user部分，即可进入账户信息管理模块，其中包括个人信息修改。

在个人信息修改模块，用户可以查看和上传自己的头像，查看和修改自己的用户名，查看和修改自己的登陆邮箱，并查看自己的注册时间。

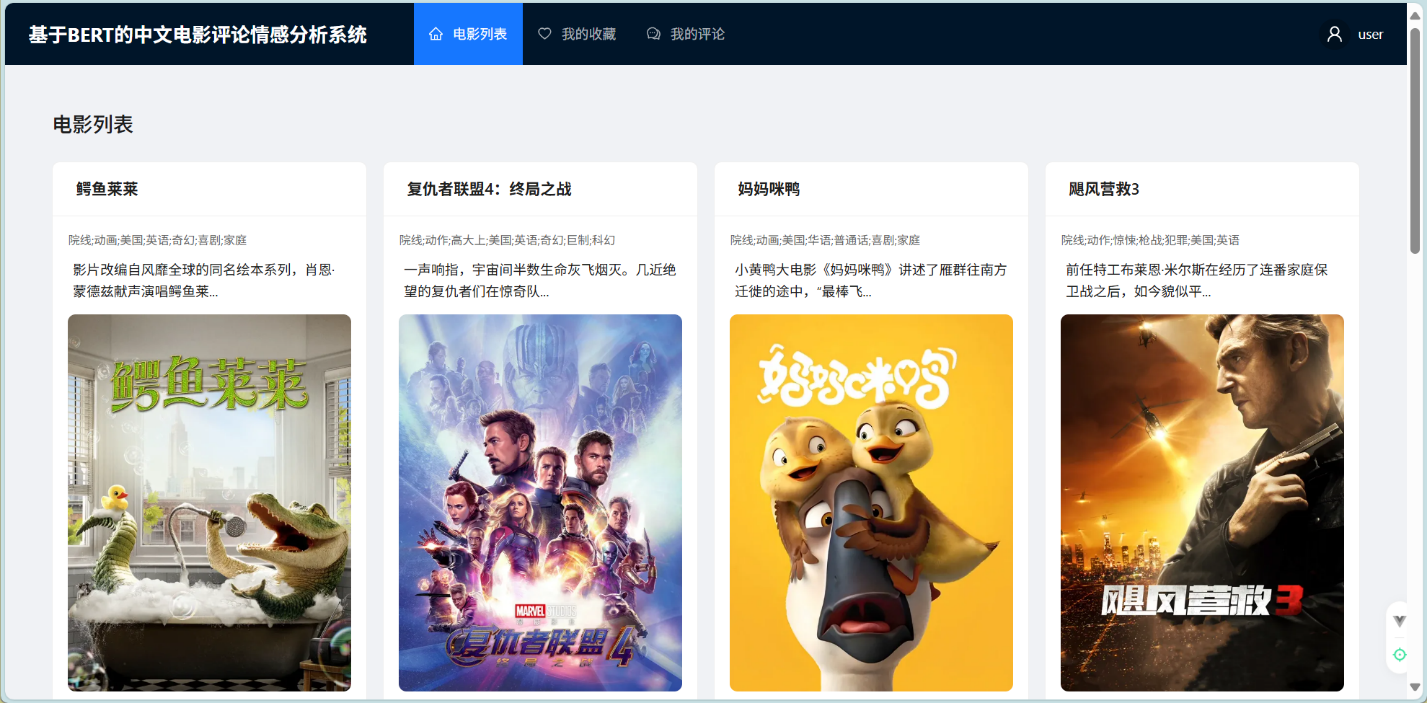


图 4‑25 普通用户系统界面

在账号设置模块，用户可以完成自己的密码修改，需要输入当前密码并输入需要修改成的新密码并确认新密码以实现密码修改功能。

* 1. 评论管理功能实现

用户在本系统当中，可以在对应的电影评论模块，发表自己的评论，如图4-26所示，同时可以在我的评论模块，查看自己已经发布的台词，同时可以删除已经发表的评论，如图4-27所示，

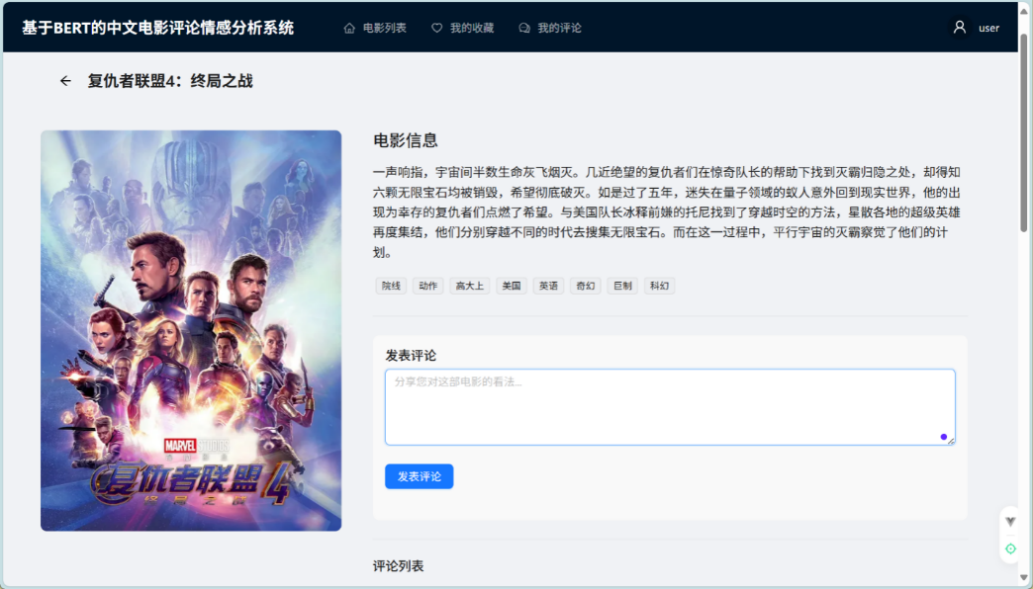


图4-26 用户发表评论功能



图4-27 用户查看和删除评论功能

管理员通过登陆管理员账户就可以进入管理系统，在本系统当中，管理员通过点击评论管理功能，管理员可以查看用户已发布的评论，评论的情感判别标签，以及可以编辑评论和删除评论，如图4-28所示：



图4-28 评论管理功能

* 1. 数据分析功能实现

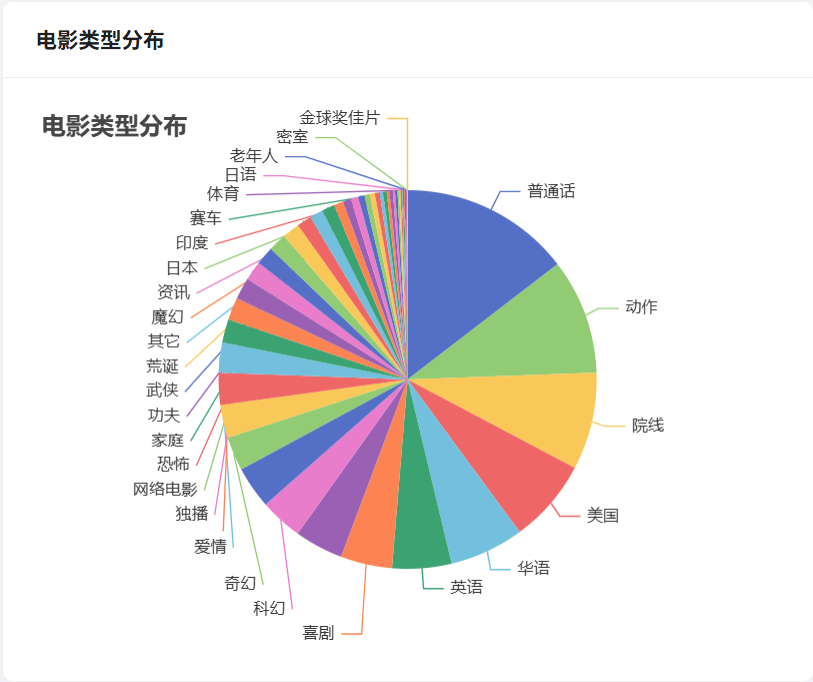
管理员通过登陆管理员账户就可以进入管理系统，在本系统当中，管理员通过点击数据分析功能，管理员可以查看系统中已上架电影的类别分布，从而帮助决定可以上架哪些类别的电影，如图4-29所示：

图 4-29 电影类别分布

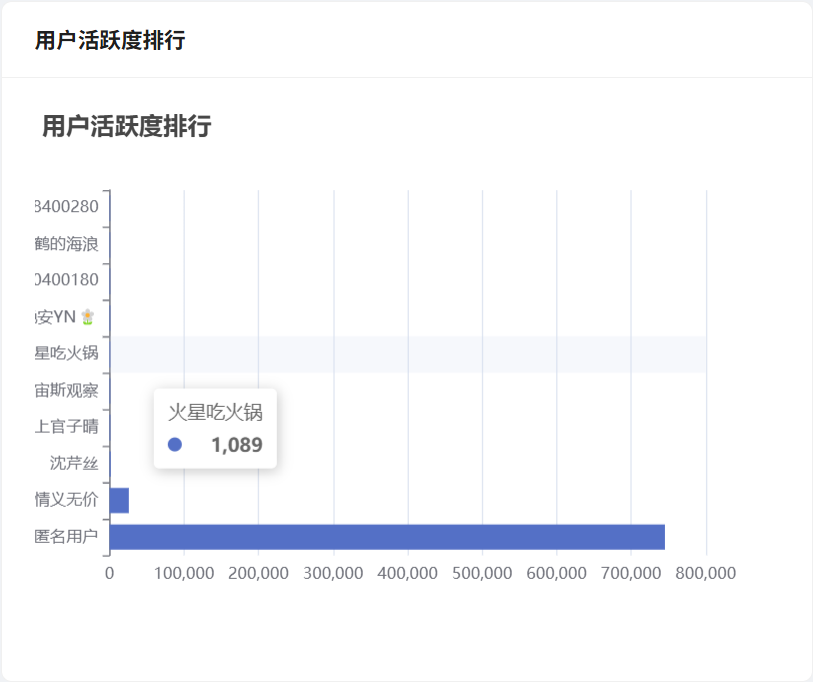
管理员通过登陆管理员账户就可以进入管理系统，在本系统当中，管理员通过点击数据分析功能，管理员可以查看用户的活跃度排行，从而了解哪些是活跃用户，便于进一步跟踪了解，如图4-30所示

图4-30 用户活跃度排行

可以快速查看到电影评分分布情况，从而可以快速看出观众对于电影的喜爱程度，如图4-31所示：

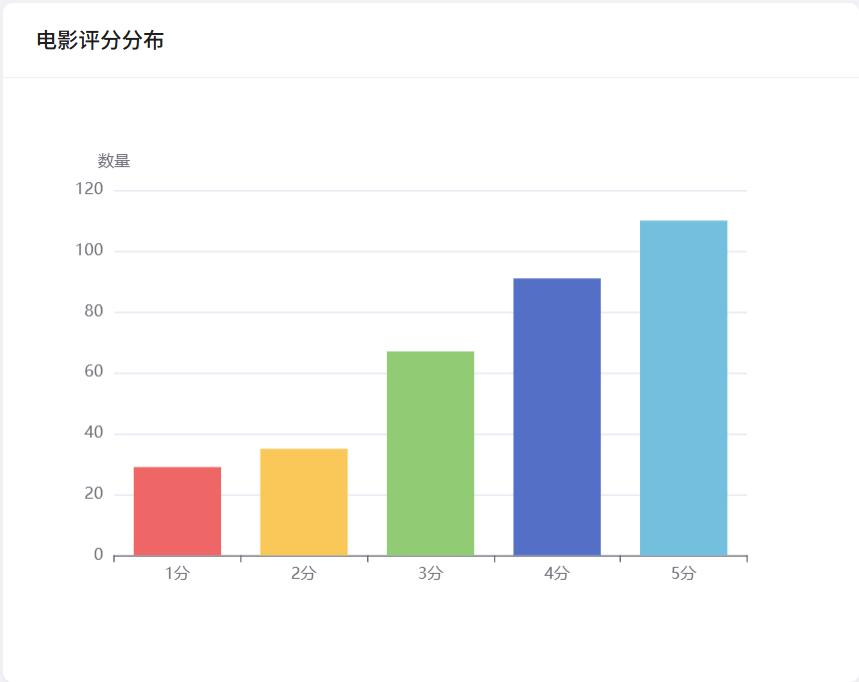


图4-31 电影评分分布

* 1. 本章小结

本章详细阐述了电影评论情感分析系统的实现过程，包括硬件和软件的开发环境配置、用户认证、电影展示、评论分析、情感分析等核心功能模块的实现、可视化交互等详细设计。系统通过Vue 3+FastAPI实现前后端分离架构，在此基础上，通过使用BERT模型并结合分类器实现了中文电影评论情感分类，结合MongoDB数据库存储分析数据，关键技术包括JWT认证、bcrypt密码加密、异步爬虫数据获取、BERT微调与评估、ECharts可视化等内容。最终系统支持用户交互、电影信息管理、评论情感分析、多维度数据展示等功能。

系统测试

本节测试均在如表5-1所示环境下完成。

表 5‑1 测试环境

|  |  |  |
| --- | --- | --- |
| CPU | GPU | OS |
| Intel ® Core(TM) i7-12650H 2.30GHz | NVIDIA RTX 4060 6G | Windows 10 64位操作系统 |

* 1. 系统部署与优化

系统部署定义了系统所用的服务，包括端口映射、环境变量、依赖关系、数据持久化等部署。此处采用docker-compose部署方式，打包好环境所需要的依赖，并创建数据库卷持久化数据

* 1. 性能优化与负载测试

性能测试主要通过使用aiohttp模拟大量并发用户去请求特定的API接口，在此基础上测量系统的响应时间、吞吐量（QPS）和错误率，以评估系统在高负载下的表现。测试结果显示，在模拟1000个用户同时请求的情况下，系统平均响应时间小于1秒， CPU占用保持在30%以下，内存占用未超过1GB，性能表现良好。

* 1. 用户功能模块测试

该部分测试关注用户基本功能的使用，包括用户注册、登陆等基本功能。如表5-1是该模块实际使用情况测试的相关内容：

表5-1 用户功能模块测试清单

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 测试描述 | 预期结果 | 实际结果 |
| Try-1-1 | 用户注册输入两次相同密码 | 不通过 | 不通过 |
| Try-1-2 | 用户注册输入无效邮箱 | 不通过 | 不通过 |
| Try-1-3 | 用户输入正确账号密码登陆 | 通过 | 通过 |
| Try-1-4 | 用户修改密码输入两次不同新密码 | 不通过 | 不通过 |

功能测试1-4：用户修改密码输入两次不同新密码

在用户账号设置模块，用户如果需要修改密码那么需要输入正确的原密码，和想要设置的新密码，如果输入的当前密码错误，则会报错当前密码错误，无法完成密码修改。同时，如果用户想要修改的新密码，如果输入的两次新密码不同，则会报错两次输入的密码不一致，从而无法完成密码修改.

* 1. 电影模块测试

该部分测试关注系统中电影模块功能是否完备，包括但不限于电影信息是否能够正常展示，用户能否收藏自己感兴趣的电影和管理员是否能够正常上架、下架和编辑电影信息。该模块实际使用情况测试相关内容如表5-2所示：

表5-2 电影模块测试清单

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 测试描述 | 预期结果 | 实际结果 |
| Try-2-1 | 用户登陆页面后能够查看电影信息 | 通过 | 通过 |
| Try-2-2 | 用户收藏电影之后可查看已收藏电影 | 通过 | 通过 |
| Try-2-3 | 管理可上架，下架，编辑电影 | 通过 | 通过 |

* 1. 评论管理模块测试

该模块主要是对电影评论进行管理，包括但不限于普通用户发表、删除自己的评论，管理员对普通用户的评论进行管理，该模块实际使用情况测试相关内容如表5-3所示：

表5-3 电影模块测试清单

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 测试描述 | 预期结果 | 实际结果 |
| Try-3-1 | 用户发表自己的评论 | 通过 | 通过 |
| Try-3-2 | 用户删除自己的评论 | 通过 | 通过 |
| Try-3-3 | 管理员进行普通用户评论管理 | 通过 | 通过 |

* 1. 情感分析模块测试

在系统中，情感分析模块主要是分析电影评论的情感类别，功能包括但不限于可以展示系统中任意一部电影的评论情感分布情况，影评随时间的发布变化，该模块实际使用情况测试相关内容如表5-4所示：

表5-4 情感分析模块测试清单

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 测试描述 | 预期结果 | 实际结果 |
| Try-4-1 | 管理员点击影评分析查看某部电影影评情感分布情况 | 通过 | 通过 |
| Try-4-2 | 管理员点击影评分析查看某部电影影评情感随时间变化情况 | 通过 | 通过 |

* 1. 数据分析模块测试

数据分析模块主要是对系统中已有的多种数据进行多类型的分析，根据不同数据的代表含义从而得到多角度的结果，包括但不限于用户活跃排行，电影评分情况，电影类别分布，影评关键词，该模块实际使用情况测试相关内容如表5-5所示：

表5-5 数据分析模块测试清单

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 测试描述 | 预期结果 | 实际结果 |
| Try-5-1 | 点击数据分析查看电影类别分布 | 通过 | 通过 |
| Try-5-2 | 点击数据分析查看用户活跃度排行 | 通过 | 通过 |
| Try-5-3 | 点击数据分析查看电影评分情况 | 通过 | 通过 |
| Try-5-4 | 点击数据分析查看影评关键词 | 通过 | 通过 |

* 1. 系统管理模块测试

在该模块当中，实现管理员账户对账户的管理，包括查看账户名，用户邮箱，用户角色和删除用户的功能，该模块实际使用情况测试相关内容如表5-6所示：

表5-6 系统管理模块测试清单

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 测试描述 | 预期结果 | 实际结果 |
| Try-6-1 | 管理员账户对账户进行查看，删除 | 通过 | 通过 |

* 1. 本章小结

本章通过系统测试验证了电影评论情感分析系统的稳定性和可用性，通过采用Docker容器化部署确保环境一致性，性能测试显示系统在并发请求下响应时间小于1秒，资源占用合理。功能测试部分覆盖用户管理、电影展示、评论分析等核心模块，验证了系统各接口的完整性，用户测试部分主要通过注册、登录、密码修改等交互，确保有提示信息能够引导，提高了系统的易用性。测试结果表明系统满足设计需求，具备良好的性能和用户体验。

总结与展望

* 1. 总结

本项目成功实现了基于BERT的中文电影评论情感分析系统，并且独立完成了从系统设计、开发到测试的整个过程。项目涵盖了前端用户界面设计、后端服务逻辑实现的一系列内容，包括实现了用户管理、电影信息管理、评论管理、数据分析等功能模块，并通过实验验证了BERT模型在处理中文电影评论情感分析任务上的性能和效果，同时也展示了该系统在实际应用中的潜力和价值。

* 1. 展望

在整个项目的开发过程中，我深刻体会到自然语言处理技术的价值和重要性，在处理信息和理解时，选择合适的算法能够事半功倍，我认识到模型训练时数据预处理、模型构建、训练调参等过程对最终处理的准确度有着很大的影响，为此在数据清洗、特征提取等前期准备工作中投入足够的时间和精力是非常必要的。这次经历让我学到了很多宝贵的知识和技能，也增强了我的独立解决问题的能力，未来我将继续学习相关知识，继续增强对计算机领域问题的解决能力。

参考文献

1. 李佳. 实现人民对美好生活向往的路径[J]. 党课参考, 2023(6): 65-70.
2. 王宜文, 张云. 互联网时代的电影评论[J]. 艺术评论, 2015(8): 14-22.
3. 司若, 黄莺, 许婉钰. 近年来我国网络影评发展脉络研究[J]. 当代电影, 2020(3): 137-143.
4. 张璐. 从Python情感分析看海外读者对中国译介文学的接受和评价：以《三体》英译本为例[J]. 外语研究, 2019, 36(4): 80-86.
5. 刘玲玉, 邓燕燕. 基于Python情感分析和批评隐喻的网络话语分析——以影片《流浪地球》中美德影评为例[J]. 江苏大学学报(社会科学版), 2022, 24(3): 76-88.
6. Pang B, Lee L, Vaithyanathan S. Thumbs up? Sentiment classification using machine learning techniques[C]//Proceedings of EMNLP. 2002.
7. Taboada M, Brooke J, Tofiloski M, et al. Lexicon-based methods for sentiment analysis[J]. Computational Linguistics, 2011, 37(2): 267-307.
8. Kim Y. Convolutional neural networks for sentence classification[C]//Proceedings of EMNLP. 2014: 1746-1751.
9. Tang D, Qin B, Liu T. Document modeling with gated recurrent neural network for sentiment classification[C]//Proceedings of EMNLP. 2015: 1422-1432.
10. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in NeurIPS. 2017: 5998-6008.
11. Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding[C]//Proceedings of NAACL. 2019: 4171-4186.
12. Zadeh A, Liang P P, Poria S, et al. Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion[R]. ACL, 2018.
13. Baccianella S, Esuli A, Sebastiani F. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis[C]//Proceedings of LREC. 2010.
14. Hutto C J, Gilbert E. VADER: A parsimonious rule-based model for sentiment analysis of social media text[C]//Proceedings of ICWSM. 2014.
15. Liu Y, Ott M, Goyal N, et al. RoBERTa: A robustly optimized BERT pretraining approach[J]. arXiv preprint arXiv:1907.11692, 2019.
16. Kim Y. Convolutional neural networks for sentence classification[C]//Proceedings of EMNLP. 2014: 1746-1751.
17. Rajpurkar P, Zhang J, Lopyrev K, et al. SQuAD: 100,000+ questions for machine comprehension of text[C]//Proceedings of EMNLP. 2016: 2383-2392.
18. Sun C, Qiu X, Xu Y, et al. How to fine-tune BERT for text classification?[C]//Proceedings of CCL. 2019: 194-206.
19. Yamada I, Asai A, Shindo H, et al. LUKE: Deep contextualized entity representations with entity-aware self-attention[C]//Proceedings of EMNLP. 2020: 6442-6454.
20. Strubell E, Ganesh A, McCallum A. Energy and policy considerations for deep learning in NLP[C]//Proceedings of ACL. 2019: 3645-3650.
21. Gururangan S, Marasović A, Swayamdipta S, et al. Don't stop pretraining: Adapt language models to domains and tasks[C]//Proceedings of ACL. 2020: 8342-8360.
22. Sanh V, Debut L, Chaumond J, et al. DistilBERT, a distilled version of BERT[C]//Advances in NeurIPS. 2019: 33.
23. Houlsby N, Giurgiu A, Jastrzebski S, et al. Parameter-efficient transfer learning for NLP[C]//Proceedings of the 36th International Conference on Machine Learning (ICML). 2019: 2790-2799.
24. Raffel C, Shazeer N, Roberts K, et al. Exploring the limits of transfer learning with a unified text-to-text transformer[J]. Journal of Machine Learning Research, 2020, 21(140): 1-67.
25. Chodorow K. MongoDB: The Definitive Guide[M]. Sebastopol: O'Reilly, 2013.
26. Banker K. MongoDB in Action[M]. Shelter Island: Manning Publications, 2011.
27. Sharma R. Mastering MongoDB Aggregation Framework[M]. Birmingham: Packt Publishing, 2021.
28. Bradshaw S. MongoDB Indexing Strategies[M]. New York: Apress, 2015.
29. MongoDB Inc. Security Reference Architecture[R]. 2022.
30. Sadalage P. NoSQL Distilled[M]. Boston: Addison-Wesley, 2012.

**声 明**

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得四川大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

本学位论文成果是本人在四川大学读书期间在导师指导下取得的，论文成果归四川大学所有，特此声明。

作者签名： 导师签名：

年 月 日

**学位论文使用授权书**

本学位论文作者完全了解四川大学有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或相关机构送交论文的原件、复印件和电子版，允许论文被查阅和借阅。本人授权四川大学将本学位论文的全部或部分内容编入有关数据库进行信息技术服务，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文，并用于学术活动。

（涉密学位论文在解密后适用于本授权书）

作者签名： 导师签名：

年 月 日

致 谢

首先