南京大學

研 究 生 毕 业 论 文
（申请工程硕士学位）

论　文　题　目　基于知识图谱的众测智能推荐系统的设计与实现

作　者　姓　名　徐佳炜

学科、专业名称　工程硕士（软件工程领域）

研　究　方　向　软件工程

指　导　教　师　陈振宇 教授，冯洋 助理研究员

2021 年 **5** 月 **18** 日

学　　号：**MF1932206**

论文答辩日期：**2021** 年 **5** 月 **20** 日

指 导 教 师：　　（签字）

The Design and Implementation of Intelligent Recommendation System for Crowdsourced Testing Based On Knowledge Graph

by

Jiawei Xu

Supervised by

Professor Zhenyu Chen, Assistant Researcher Yang Feng

A dissertation submitted to
the graduate school of Nanjing University
in partial fulfilment of the requirements for the degree of
MASTER OF ENGINEERING
in
Software Engineering

Software Institute
Nanjing University

May 18, 2021

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：基于知识图谱的众测智能推荐系统的设计与实现

工程硕士（软件工程领域）专业 2019 级硕士生姓名：徐佳炜

指导教师（姓名、职称）：陈振宇 教授，冯洋 助理研究员

摘　要

众包测试作为飞速发展的一种众包模式，能够线上招募众测工人，无需准备测试环境，获得大量缺陷报告和测试用例，成本低，速度快，得到了企业和个人的青睐，但也因为众包工人身份各异，测试水平高低不一，众测平台收集数据与反馈速度慢，出现了重复报告数量高，测试页面覆盖率低，缺陷报告质量差，众包工人积极性差的问题，给众包测试的实际应用带来了一定的挑战。

本文针对众包测试中的上述问题，提出了基于知识图谱的众测智能推荐技术，开发了众测智能推荐系统。在众包测试的过程，众包工人提交测试报告相关信息，系统获取报告数据和工人信息，进行知识图谱的构建。然后使用词向量模型进行关键词的特征学习，使用知识图谱特征学习模型进行实体和关系的特征学习。接着，系统将特征作为推荐系统的辅助信息引入，使用推荐算法向众包工人推荐缺陷报告，众包工人可以进行查看，审核，克隆和优化。最后，系统使用相似度算法进行缺陷报告重复性检测，将重复的缺陷报告置为零分。

本系统使用 Angular2 作为众测平台前端框架，SpringBoot 作为众测平台后端框架，Django 作为推荐和重复性检测服务框架，保证系统的稳定性。使用 Redis 作为缓存机制，MongoDB 作为数据库，Nginx 实现反向代理和负载均衡，提高系统的可用性和可扩展性。使用 Docker 容器技术对服务进行虚拟化，实现服务间的松耦合。

本系统提供了众包测试过程中的推荐服务和众包测试结束后的重复性检测服务，通过及时的反馈和信息交流提高了众包工人的积极性，减少了重复报告的产生，提高了报告质量和测试页面覆盖率，满足了企业和个人的业务需求。本系统已落地于真实公司中的众包测试平台，经过数十场真实众包测试的检验，为企业和众包工人提供稳定可靠的众包测试服务。

关键词：众包测试，推荐系统，知识图谱，相似性检测

南京大学研究生毕业论文英文摘要首页用纸

THESIS: The Design and Implementation of Intelligent Recommendation System for Crowdsourced Testing Based On Knowledge Graph

SPECIALIZATION: Software Engineering

POSTGRADUATE: Jiawei Xu

MENTOR: Professor Zhenyu Chen, Assistant Researcher Yang Feng

Abstract

As a rapidly developing crowdsourcing model, crowdsourced testing can recruit crowdsourced workers online and obtain a large number of defect reports and test cases without preparing test environment. It is favored by enterprises and individuals because of its low cost and fast testing speed. But at the same time, because of the different identities of crowdsourced workers, different testing levels, slow data collection and feedback speed of crowdsourced platform, there are some problems, such as high number of repeated reports, low coverage rate of test pages, poor quality of defect reports, and poor enthusiasm of crowdsourced workers, which bring some challenges to the practical application of crowdsourced testing.

Aiming at the above problems in crowdsourced testing, this paper proposes an intelligent crowdsourced recommendation technology based on knowledge graph, and develops an intelligent crowdsourced recommendation system. In the process of crowdsourced testing, crowdsourced workers submit data of test reports. The system obtains report data and rowdsourced workers' information to construct knowledge graph. Then, the system uses word vector model to learn features of keywords, and uses knowledge graph feature learning model to learn features of entities and relations. Next, the system introduces the knowledge graph as the auxiliary information of the recommendation system, and recommends the defect report to the crowdsourced workers by using recommendation algorithm. The crowdsourced workers can check, review, clone and optimize the recommended reports. Finally, the system uses similarity algorithm to detect the repeatability of defect report, and sets the score of duplicate defect report to zero.

To ensure the stability of the system and the efficiency of development, this system uses Angular2 as the front-end framework, Springboot as the back-end framework of the crowd testing platform, Django as the framework of recommendation and repeatability testing service. In order to improve the performance and scalability, this system uses Redis as the caching middleware, Mongodb as the database, and Nginx as the reverse proxy mechanism to achieve load balancing. The Docker container technology is used to virtualize services and realize loose coupling between services.

This system provides recommendation service in the process of crowdsourced testing and repetitive testing service after crowdsourced testing. Through timely feedback and communication, it improves the enthusiasm of crowdsourced workers, reduces the generation of repetitive reports, improves the quality of reports and the coverage of test pages and satisfies needs of enterprises and individuals. The system has been implemented in the crowdsourced test platform of real enterprise, and has been tested by dozens of real crowdsourced testings, providing stable and reliable crowdsourced testing services for enterprises and crowdsourced workers.

keywords: Crowdsourced Testing,Recommendation System,Knowledge Graph,Similarity Detection

目　　录

插图清单

附表清单

第一章　引言

1.1　项目背景和意义

众包于 2006 年被 Howe Jeff 提出 [1]，是一种企业将任务发布在互联网，通过经济或经验促使感兴趣的人自愿参与 [2]，将过去由员工进行处理的任务承包给非特定人员的组织模式 [3]。目前，众包概念已成功应用于自然语言处理，数据标注，任务分发等各个领域，如蜂鸟众包，美团众包，京东众包等。

众包测试将众包的思想应用于软件测试领域，主要角色包括任务发起者、众测平台与众包工人 [4]。众测平台为任务发起者和众包工人提供服务，任务发起者发布测试目标，相关要求，奖励机制到众测平台，众包工人注册后获得平台通知和推荐，并自愿选择感兴趣的任务执行，执行内容包括对测试目标进行测试，创建相关的缺陷报告并提交。众测平台组织评审专家对缺陷报告进行审核，根据审核结果对众测工人进行奖惩。然后使用自动化或人工手段进行缺陷报告整编，最终形成完整的测试报告，包括测试用例和缺陷报告，满足任务发起者的测试需求 [5]。

众包测试的优势在于可以在短期内招募大量人员进行测试，能够模拟各个地理位置和测试环境，测试设备，测试周期短，成本低，效率高。但众测同时也有异地参与，众包工人测试水平不一，约束工人能力差，任务分配不均等问题，给众包测试在工业界的实际应用带来一定的挑战 [6]。

众包测试具有测试速度快，测试成本低的特点，因此受到公司和个人的青睐 [7]，出现了许多面向企业和个人的众测平台。在目前的众包测试和众测平台中，众包工人接受任务后对测试目标进行测试，创建缺陷报告并进行提交，由于众包工人测试水平高低不一，众测平台收集数据与反馈不及时，众包测试往往会出现以下问题，以工业界真实众测平台为例进行说明，以下简称 M 平台：

1）重复报告数量高 [8]：由于在众测过程中，众包工人之间无法进行交流，也不知道他人的进度与成果，众包工人会发现大量重复 Bug，提交大量重复报告，这些报告不仅浪费了众包工人的时间和精力，也给最后报告的评审和整编交付带来困难。据统计，M 平台每场众包测试有 30% 左右的报告为重复报

告，大部分为“登录失败”，“按钮失效”等简单且明显的 Bug。

2）测试页面覆盖率低 [9]：同样，由于无法同步信息，在一定的众测时间内，众包工人发现的问题往往集中于几个易于发现 Bug 或者逻辑上先被测试的页面，有许多页面长时间处于无人问津的状态，直到众测结束也没有工人踏足，许多潜在 Bug 无法被挖掘。据统计，M 平台大约有 27% 的众测任务直到任务结束都没能实现测试页面全覆盖。

3）缺陷报告质量差 [10]：由于部分众包工人测试或语言组织能力较差，提交的缺陷报告描述不清晰，不完整，不深入，质量较差，需求方获得报告后难以定位和复现 Bug，对于这样的缺陷报告，评审和整编都较为困难。据统计，M 平台每场众包测试平均有 25% 的缺陷报告被评审为 0 分或低分（6 分以下，10 分满分），原因包括语言描述不清晰，没有截图说明等。

4）众包工人积极性差 [11]：由于没有及时的反馈和竞争机制，部分众包工人积极性较差，在测试过程中浅尝辄止，发现几个明显的 Bug 后便停止测试，不愿意投入过多精力进行尝试，发掘潜藏较深的 Bug。据统计，截止到 2021 年 3 月底，M 平台共有 2542 人次参与了 100 场众包测试，提交了 16920 份缺陷报告，平均每人次提交 7 份，大部分众包工人只提交了 1-3 份，对于一场至少两小时的众包测试任务来说，提交数量明显不足。

针对以上问题，本文提出基于知识图谱的众测智能推荐技术，开发了众测智能推荐系统。在众测任务执行阶段，通过搜索推荐的方式让众包工人了解其他人的进度和提交情况，查看具体缺陷报告信息，从源头减少重复报告的产生，提高测试页面覆盖率；进行报告审核和优化，利用群体智能提高缺陷报告质量；通过及时的反馈和竞争机制，提高众包工人的积极性。在众测任务评审阶段，进行缺陷报告的自动化查重，将重复报告标出，减轻评审压力。最终，本系统为众包测试中交付报告的质量提高做出贡献，也是知识图谱与缺陷报告推荐相结合的一次实践，为相关科研工作提供验证和参考。

1.2 国内外研究现状

1.2.1 众包测试

众包测试是众包软件工程的子分支，章晓芳等定义“支持软件测试的所有众包活动，包括所有众包方法、技术、工具和平台”都属于众包软件测试 [5]，认为众包测试可以对现存的测试方法和测试流程进行一定程度的优化 [5]，主要包

括针对特定测试类型的优化和测试流程的优化。

测试类型优化主要包括以下几个方面：众包 QoE 测试可以快速收集不同地区和设备的数据，降低测试成本，缩短测试时长 [12]；相比传统方式，众包可用性测试可以降低测试成本，通过在不同地区和环境进行测试得到来自世界各地的真实数据 [13]；与传统实验室测试方法相比，众包 GUI 测试更有利于 GUI 测试的持续开展，可以有效补充传统方法，提高测试效果 [14]；众包性能测试能够快速收集不同用户的行为和测试设备数据，从各个角度和使用方式发现软件的性能问题，并协助开发团队进行改进和优化 [15]。测试流程优化主要包括以下几个方面：众包测试用例生成通过人工创建测试用例，能够获得比自动测试用例生成技术更高的代码覆盖率 [16]；众包程序调试与修复引入众包测试信息作为程序调试的辅助信息，进行缺陷报告生成，缺陷定位和修复者推荐 [17]；软件开发者可以通过众包软件评估获得众包工人的反馈，从他人的视角看待软件，更好的评估自己开发的软件系统 [18]。

随着众包测试的应用领域不断扩张，国内外也出现了许多相关的平台。国内的众测平台包括腾讯 QQ 众测[①]，百度众测[②]，Testin 云测[③]等；国外的众包平台包括 CrowdFlower[④]，AMT[⑤]等。这些众测平台往往因为母公司如腾讯，百度的原因拥有较多的用户量和流量，但平台规范与众测流程还处于起步阶段，并不能将庞大的用户量转化为自身的测试能力，发展为优秀的商业化众测平台，为广大有需求的企业提供众包测试解决方案。大部分平台都有任务类型单一，任务量少，奖惩机制差的问题，部分平台甚至进行自我限制，以腾讯 QQ 众测为例，任务仅仅针对腾讯集团旗下的产品进行测试，奖励物品为 Q 币，购物卡，公仔等，众包工人积极性不高。在提交缺陷报告的过程中，系统反馈与交互能力较差，采取竞争式的众包模式 [19]，众包工人单独完成测试任务，无法发挥群体智能的能力。测试结束后审核任务繁琐，重复报告多，测试覆盖率低，测试报告质量差，与传统的自动化脚本测试相比，并没有体现出众包测试应该有的优势。

虽然众包测试近年来迅速发展，但协作式众包作为众包测试未来的发展方向，相关平台的研究较少，如何快速完成测试，降低测试成本，提高测试报告质量，仍是需要解决的问题。本文希望将基于知识图谱的众测智能推荐技术应

[①] https://task.qq.com/
[②] http://test.baidu.com/
[③] https://www.testin.cn/
[④] https://appen.com/
[⑤] https://www.mturk.com/

用于协作式众包测试平台，在众测执行阶段推荐缺陷报告，让众测工人进行查看，审核，克隆和优化，提高工人的积极性，减少重复报告，提高测试页面覆盖率和报告质量；在众测任务结束后进行自动查重，减少报告评审和整编的难度，并在真实环境下进行验证，优化并解决目前众测平台面临的一些问题。

1.2.2 知识图谱

知识图谱由 Google 首先提出，随着信息爆炸的时代来临，人们不仅关注信息本身，还要考虑信息之间的联系，信息是知识的基础，知识是人主观上对信息的抽象总结和归纳。知识图谱以多关系图的形式对知识进行表示，目前已被广泛应用于搜索推荐，大数据风控，金融投资，医疗卫生等领域，成为人工智能领域的关键技术之一 [20]。

随着信息时代来临，大量数据被共享，国内外研究人员构建了大量知识库，主要分为开放链接知识库和垂直行业知识库。开放链接知识库覆盖大量领域，属于通用知识库 [21]，主要包括 Freebase[22]，Wikidata[①]，DBPedia[23]，YAGO[24] 等。垂直行业知识库描述特定的领域，需要一定的领域知识，应用范围也较为局限，但在特定领域的表现较好，主要包括 IMDB[②]，MusicBrainz[③]，ConceptNet[④]等，本文构建的知识图谱也是垂直行业知识库，主要应用于缺陷报告领域。

知识图谱可以被应用于多个领域，在智能搜索领域，系统首先对用户的请求进行语义理解，然后在知识库中进行检索，得到结果，并以卡片的形式进行展示，国内应用如百度，搜狗，国外应用如谷歌，必应；在深度问答领域，大部分问答系统将问题进行分解，对小问题的答案进行整合，得到结果，如天猫精灵，Siri 等；社交网络方面，Facebook 等社交媒体使用知识图谱进行好友推荐，满足用户寻找相似的人的需求。垂直行业中，金融行业将知识图谱应用于大数据风控；电商行业使用知识图谱进行商品信息的查找和相关信息的展现，提高用户体验；还有更多的行业需要使用知识图谱进行信息整合和关联，更便捷地获取领域知识。

[①] http://www.wikidata.org/
[②] https://www.imdb.com/
[③] https://musicbrainz.org
[④] http://conceptnet5.media.mit.edu

1.2.3 推荐系统

随着互联网的不断发展，每个人接收的信息呈指数级爆炸，如何在众多信息中进行检索，成为了热点问题，推荐系统根据用户喜好，向用户提供精准有效的信息推荐服务，如电商行业的“猜你喜欢”，短视频行业的推荐信息流等。

传统的推荐方法包括基于内容的推荐，Han 提出了基于相似性的缺陷报告推荐，使用 WMD 算法进行文档相似度计算，向工人推荐与正在编辑的报告相似的缺陷报告；协同过滤推荐，基于用户与系统的交互进行推荐，但存在稀疏性和冷启动的问题，Han 提出了基于协同过滤和 PMF 的测试任务推荐方法；混合推荐方法将推荐系统与辅助信息融合，提升推荐效果，如本文使用知识图谱作为辅助信息。

基于机器学习的推荐方法主要包括基于深度神经网络的推荐方法、基于卷积神经网络的推荐方法、基于循环神经网络和长短期记忆神经网络的推荐方法、基于图神经网络的推荐方法 [25]，能够更好的进行用户和物品的特征提取，有较强的抗干扰能力，提高了推荐的效果。

1.3 本文主要工作

针对以上研究现状和目前众包测试平台所面临的问题，本文从众包测试平台的角度出发，提出了基于知识图谱的众测智能推荐技术，开发了众测智能推荐系统，实现众测任务执行阶段的报告推荐，信息共享，群体智能和众测任务评审阶段的重复性检测。目前本系统已落地于真实众包测试平台，为众包工人和管理员提供稳定高效的报告推荐和重复性检测服务。

众包工人在众测任务执行阶段，可以进行测试报告，测试用例，缺陷报告的提交，同时可以通过主动搜索和被动推荐获得他人提交的缺陷报告信息并查看，从源头减少重复性报告的产生，提高测试效率，减轻评审专家的工作量，提高最终交付报告的质量，

众包工人可以通过对他人报告进行审核对报告质量做出评价，也可以进行报告克隆并在此基础上进行修改和提交，优化报告质量。用户的行为信息和协作数据均会被系统记录，对报告的质量和优先级做出影响，也会反作用于推荐系统的推荐方式。

虽然在众测任务执行阶段通过系统推荐和群体智能的方法减少重复缺陷报告的产生，但这仍是不可避免的。因此众测任务结束后，众测管理员可以进行

缺陷报告重复性检测，筛选出重复率过高的缺陷报告，在众测管理员人工复核后将重复报告的参考分置为零分，减少评审专家和整编报告的工作量。

最后对系统进行测试，保证了系统的可用性，可靠性和基于知识图谱的众测智能推荐技术的有效性，证明了系统可以满足企业和众包工人的业务需求，提供一定的使用价值。

1.4 本文组织结构

本文首先论述了众包测试目前的发展情况和遇到的问题，说明了众包测试和知识图谱推荐技术的国内外研究现状。然后介绍了完成本系统所需要的相关工程技术和理论基础。接着分析了系统的需求，采用多种方法进行了系统整体设计和详细设计。然后对各个模块具体实现和关键代码进行了说明，并对系统进行了多种测试。最后，对本文进行了总结，说明了技术和系统实现的不足之处，对未来的技术发展和系统实现做出了展望。本文的组织结构如下:

第一章，引言。首先介绍了众包测试的发展现状，然后对众包测试，知识图谱和推荐系统的国内外研究现状进行了说明，针对上述情况，本文提出了基于知识图谱的众测智能推荐技术。

第二章，相关技术综述。对于系统中使用到的前后端技术框架，开源工具以及算法的理论基础进行了简单的介绍。

第三章，系统的需求分析与设计。首先说明了对系统的需求进行了分析，从用例，系统架构，模块划分，“4+1”视图等多个角度对系统进行了说明。然后对系统的实体类和数据库进行了设计。接着对系统使用到的推荐和重复率检测算法进行了设计，最后使用流程图和核心类图对系统的四个模块进行了详细设计和说明。

第四章，系统的具体实现。采用顺序图，关键代码，界面截图，算法说明相结合的方式，对知识图谱构建模块，知识图谱特征学习模块，缺陷报告推荐模块，缺陷报告重复性检测模块的具体实现进行了论述。

第五章，系统的测试和实验分析。通过对系统进行测试，验证了系统的可用性，可靠性和基于知识图谱的众测智能推荐技术对众测平台的提升效果，说明了系统的实用价值。

第六章，总结和展望。对论文所完成的工作进行了总结说明，针对系统的不足进行了分析，对众包测试推荐系统的未来技术发展方向作了展望。

第二章　相关技术概述

2.1　知识图谱相关技术

2.1.1　知识图谱

本系统使用知识图谱获取缺陷报告的语义信息和潜在关联，通过知识图谱特征学习和推荐系统相结合，实现缺陷报告的推荐服务。知识图谱由 Google 首先提出，以三元组进行通用表达，三元组的基本形式为（实体 1，关系，实体 2）[21]。实体是知识图谱中最基本的元素，可以用一个全局唯一的 ID 进行表示，以地理知识图谱为例，知识图谱中可以存在“中国”，“美国”，“亚洲”等实体；关系代表实体之间的联系，可以为不同的实体之间定义不同的关系，只要符合逻辑或真实世界的情况即可，如“中国”与“亚洲”之间的关系可以是“位于”，“江苏”与“安徽”之间的关系可以是“接壤”。概念主要描述实体的类型，如“中国”的概念可以是“国家”；属性主要指实体具有的特征，如“中国”有属性“面积”，这个属性的值为“960 万平方公里”。

将知识图谱与推荐系统结合，可以引入更多的语义关系，发现用户更深层次的兴趣，提高推荐系统的精准度；定义更多的关系类型，有利于推荐结果的发散，提高推荐系统的多样性；通过将用户的历史行为数据和推荐结果相关联，使得推荐结果可解释，提高用户对推荐结果的满意度 [26]。

知识图谱数据一般来源于业务领域的数据，如地理，历史等知识领域，或者网页公开数据，如维基百科等。本系统构建的知识图谱的数据主要是众测工人和报告数据，来源于真实众测任务场景下众测工人信息和工人提交的报告，经过筛选后，本身具有真实性和有效性。缺陷报告作为文档化数据具有丰富的语义信息和潜在联系，适合使用图结构描述各个实体本身的属性以及它们之间的关系，因此采用知识图谱作为众测智能推荐技术的理论支持。

2.1.2 知识图谱存储技术

知识图谱构建完成后，需要进行持久化的存储，主要包括三种方法：关系型数据库存储，图数据库存储和 RDF 存储。

关系型数据库存储包括三元组表存储，属性表存储和垂直分割存储，存储开销大。在表构建结束后进行修改极为麻烦，多级查询和推理实现复杂。

图数据库存储使用图结构进行知识图谱的表示，图中节点代表实体，边代表关系，节点属性代表实体属性。图数据库在查询速度上要优于关系型数据库，多级查询性能较好，但更新较为复杂，分布式存储难度较高。

RDF（Resource description framework）存储 [27] 采用空间换时间的策略，通过六重索引对三元组搜索的效率问题进行了优化，其语义表达能力强，但因索引众多，存储开销巨大。

考虑到存储开销和使用效率，本系统使用非关系型数据库 MongoDB 和图数据库 Neo4j[28] 进行数据存储。本系统存储的主要内容是测试报告，缺陷报告等文档化数据，MongoDB 面向集合，对于文档存储的支持较好，且支持 Json，因此将大部分数据存储至 MongoDB。Neo4j 作为主流的图数据库，天然能够对知识图谱进行很好的表示，部署简单，使用方便，具有完善的查询语言和极高的查询效率，因此将推荐相关的实体及关系数据存储至 Neo4j，保持知识图谱的轻量级和专业性，提高复杂查询的效率。

2.1.3 知识图谱特征学习技术

知识图谱特征学习方法包括基于嵌入的方法和基于路径的方法，基于嵌入的方法通过图嵌入对实体和关系进行表征，进而扩充原有物品和用户表征的语义信息 [29]。基于路径的方法对物品，用户之间的链接关系进行挖掘，构造推荐算法。本文使用知识图谱中实体和关系的特征作为推荐系统的辅助信息，因此选用基于嵌入的方法。

基于嵌入的方法能够学习知识图谱中实体和关系的特征，得到向量表示，主要包括 Trans 系列和基于异质信息网络的图嵌入方法。Bordes 等人提出了 TransE 模型 [30]，将关系 r 看做是从三元组头节点 h 到尾节点 t 的翻译，定义了一个距离函数 $d(h + rt)$ 来衡量 $h + r$ 和 t 之间的距离，通过训练最小化距离函数，得到向量表示。在 TransE 模型的基础上，Wang 等人提出了 TransH 模型 [31]，对于关系 r，由一个超平面 W_r 和一个关系向量 d_r 表示，而不是和实体在

同一个嵌入空间，提高了对复杂关系的建模效果。Lin 等人在前两种模型的基础上提出了 TransR 模型 [32]，对于实体的每一个关系，在单独的关系空间中对实体和关系进行建模，并在关系空间中进行头尾实体的翻译。Ji 等人在 TransR 模型的基础上提出了 TransD 模型 [33]，将头尾实体分别投影到关系空间得到投影矩阵，计算头尾实体的投影向量，减少了模型参数，降低了计算的时间复杂度，更好地应用于大规模知识图谱地特征学习。

异质信息网络方面，Yang[34] 等利用 Metapath2Vec[35]，通过元路径下的随机游走方式构造邻居节点集合，并基于 skip-gram 模型对实体进行表示。Palumbo[36] 等基于 node2vec[37]，提出了一种针对异质信息网络图的实体向量表示方法。

本系统的知识图谱数据量较大，同时 Trans 系列与推荐系统的结合相对更加成熟，因此选用 TransD 模型进行知识图谱中实体和关系的特征学习。

2.1.4 知识图谱推荐技术

知识图谱特征学习与推荐系统相结合主要包括三种方式：依次学习，联合学习和交替学习。Wang[38] 等人使用依次学习的思想，将知识图谱特征引入 CTR 中，提出 CNN 和 Attention 相结合的 DKN 模型，DKN 结合用户点击的历史新闻判断用户是否会点击新的新闻，侧重于新闻推荐场景。Zhang[39] 等人使用联合学习的思想，通过将结构信息，文本信息，图片信息融合到 CF 中，使用协同过滤实现商品推荐，提出了 CKE 模型，在电影和图书推荐上取得了较好的效果，但该模型需要使用大量在实际推荐中不易获得的额外信息，局限性较大。同样，Wang[40] 等人将知识图谱作为辅助信息引入 CTR/TopK 推荐，提出 RippleNet，在刻画用户时考虑偏好传播时的外层物品，以用户点击过的历史物品向量集合作为用户特征，可解释性很强。Wang[41] 等人使用交替学习的思想，提出 MKR 模型，推荐系统中的物品和知识图谱中的实体实际上是对同一个对象的两种描述方法，推荐模块和知识图谱特征学习模块可以通过交叉特征共享进行信息交换，获得额外信息，弥补自身不足，提高推荐的准确度。

本众测平台的数据较多，构建的知识图谱较大，进行一次知识图谱特征学习所花费的时间较长，而众测平台一个月一般进行二至三次众包测试，知识图谱数据更新频率较低。因此本文使用依次学习，使两者相互独立，通过一次特征学习得到的实体和关系向量可用于推荐系统的多次更新和训练，而知识图谱特征的更新可以在一个较长的时间间隔内进行。

2.2 相似度检测相关技术

2.2.1 Word2vec

本系统使用 Word2vec 模型进行词向量的获取。Word2vec 是一种词嵌入模型，于 2013 年被谷歌提出，主要包括两个语言模型，分别为 CBOW 词袋模型和 Skip-gram 模型。词袋模型（Bag-of-words model）不考虑文法以及词语的顺序，以一个词语的上下文作为输入来预测这个词语本身 [42]。Skip-gram 模型可以跳过某些符号，以一个词语作为输入来预测它周围的上下文 [43]。

Word2vec 采用层次 softmax 和负采样的方式来加速神经网络的训练，层次 softmax[44] 使用哈夫曼树代替二分类，将 N 分类问题转化为 $log(N)$ 分类问题，降低了问题的复杂度；负采样 [45] 通过不考虑出现概率低的词语，降低词袋规模，将预测总体类别变为预测总体类别的一个子集。

Word2vec 的应用十分广泛，既可以用来解决词语的分类、聚类和相似度计算等词语级问题，也可以作为其他任务的前置处理，进行更高级的应用，比如句子，文档的相关问题。在工业界也广泛应用于社交网络推荐，商品相似度度量，物品快速检索等领域。本文需要进行词向量的获取和短文本相似度的度量，实现缺陷报告推荐和重复性检测服务，Word2vec 模型在这两个方面都有优秀的表现，因此选用该技术。

2.2.2 句子相似度计算

句子向量的获取主要包括算术平均方法，对句子中所有词向量计算算数平均作为句子向量；基于 TF-IDF 的加权平均算法 [46]，以句中每个词的 TF-IDF 做为权重，对所有词向量加权平均作为句子向量；基于 SIF 的加权平均算法 [47]，以平滑倒词频作为权重，对所有词向量加权平均作为句子向量；WMD（Word Mover's Distance）算法 [48]，基于词移距离计算句子的相似度。

参考 Comparing Sentence Similarity Methods[①]可知，基于 SIF 的加权平均算法在大多数语料中的表现较好，因此本文选用该方法进行句子向量的获取，最后使用余弦距离进行句子相似度的计算。

[①]http://nlp.town/blog/sentence-similarity/

2.3 相关工程技术

2.3.1 Docker

本系统使用 Docker[49] 进行服务的虚拟化和集群部署。在日常开发中，常常会因为开发环境与部署环境的差异导致代码上线后出现开发环境中不存在的问题，或者需要在新的机器上运行时，先要进行很多运行环境的安装，给开发和部署带来不便。通过 Docker，开发者可以将应用及其依赖包打包进行虚拟化 [50]，使得项目自带运行环境，能够在不同机器上实现一键部署。与传统的虚拟机技术不同，Docker 容器之间实现内核共享，通过进程进行区分，能够更好地利用计算机的资源 [51]。

如图 2-1所示，Docker 的核心组件包括 Docker Client、Docker Daemon、Docker Registry Docker Container。其中 Docker Client 负责与 Docker Daemon 进行连接，发出命令。Docker Daemon 负责处理命令，主要包括创建、运行、保存容器等任务。Docker Registry 负责对 Docker 镜像进行存储和管理。Docker Container 是从景象创建而来，是运行中的镜像实例。

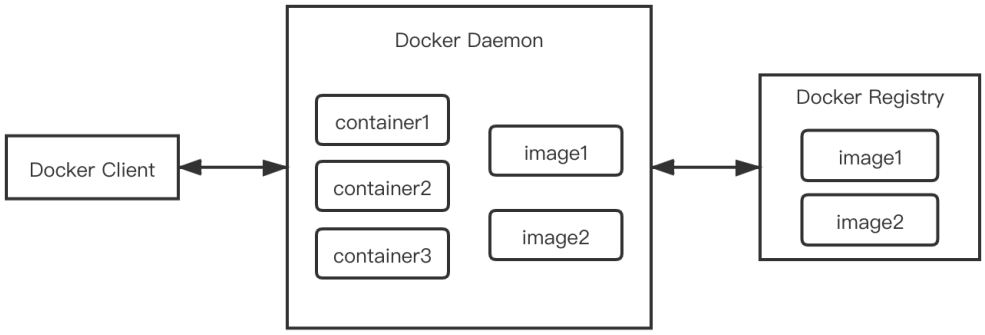

图 2-1: Docker 架构图

通过 Docker 进行代码的开发，测试，交付和部署，可以减少因开发环境，测试环境和生产环境之间的差异带来的问题。Docker 也是最适合实现微服务的机制，使用 Docker 进行集群部署，即使某个容器出现问题，也不影响系统整体的可用性，还可以通过运维系统进行故障的快速发现和修复 [52]。

本系统的后端服务包括使用 Java 语言和 SpringBoot 框架进行开发的众测后端服务，主要提供创建测试报告，查看更新删除等能力；使用 Python 语言和 Django 框架进行开发的推荐服务，主要提供缺陷报告推荐和重复性检测服务，

因此需要进行业务逻辑的解耦；同时，本系统的 MongoDB 采用主从模式，需要进行集群化部署，因此选用 Docker 进行服务的部署和运维。

2.3.2 Redis

本系统使用 Redis[53] 作为缓存机制。与传统的关系型数据库不同，Redis 将数据存储在内存中，不需要与物理磁盘进行交互，这极大地提高了数据查询和更新的速度；相较于 Memcached 等键值存储系统，Redis 还支持周期性地将数据进行持久化保存，启动后可以加载最新的持久化数据进行数据恢复和初始化，减少因系统崩溃等问题导致的数据丢失。

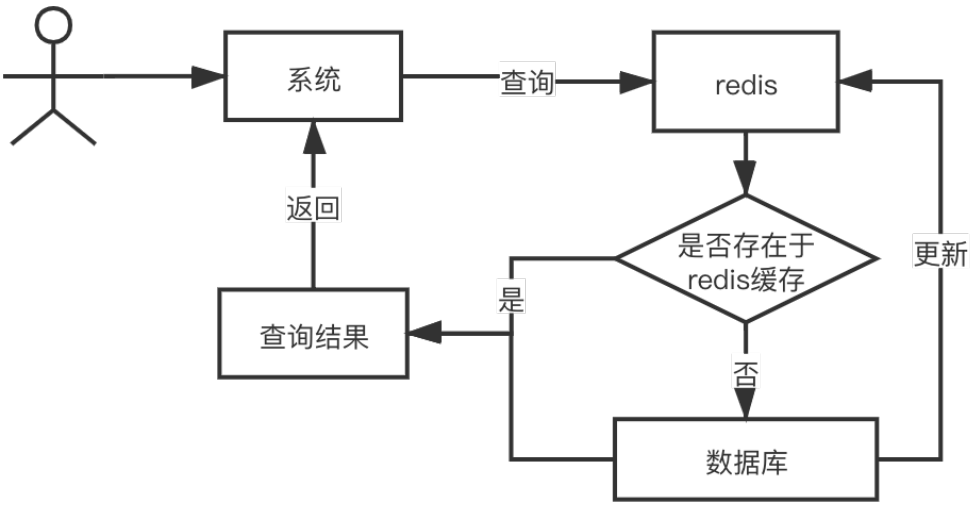

图 2-2: Redis 架构图

如图2-2所示，对于数据的查询请求，系统首先会到 Redis 中进行数据的查询，如果不存在，才会到数据库中进行查询，并将结果更新至 Redis，下次查询可以直接从 Redis 中获得结果，不必对数据库进行访问。因此，Redis 适用于不频繁变动的数据，热点数据和需要进行大量计算或查询得到的中间数据的存储，能够减轻数据库的查询压力，提高系统的响应速度和并发能力。

Redis 可以支持多种类型的数据结构 [54]，如字符串，散列，列表，集合，有序集合等，在此基础上可以很方便地实现多种功能，如共同关注，朋友圈等；可以通过自动分区和分布式部署实现高可用性和可靠性；通过持久化机制减少数据的丢失。本系统存在大量查询操作，数据类型多样且变动不频繁，热

点数据和计算中间过程数据较多，因此采用 Redis 作为本系统的缓存机制，减轻数据库的压力，提高系统的并发性。

2.4 本章小结

本章主要介绍了基于知识图谱的众测智能推荐技术所使用的工程框架，开源工具和理论基础。首先对知识图谱相关技术进行了介绍，包括知识图谱存储技术，知识图谱特征学习技术，知识图谱推荐技术；然后对相似度检测相关技术进行了介绍，包括词向量模型 Word2Vec 和句子相似度计算技术；最后对系统实现的相关工程技术进行了介绍，包括部署运维工具 Docker 和数据缓存机制 Redis，强有力地说明了这些技术的优势和选用这些技术的原因，为系统的实现提供了理论依据和基础。

第三章　基于知识图谱的众测智能推荐系统的需求与设计

本系统在众包测试任务过程中向众测工人提供信息共享，群体智能和报告推荐的功能，在众包测试任务结束后向众测管理员提供缺陷报告重复性检测的功能，目的是减少重复报告数，加快测试页面覆盖速度，提高众包工人积极性，减轻报告评审和整编的工作量，提高交付报告质量。

在一场众包测试的过程中，一个众包工人对一份测试报告进行填写，在一份测试报告中可以创建多个测试用例和缺陷报告，一个测试用例下可以发现多个缺陷，一份缺陷报告就代表工人对于发现的一个 Bug 的描述。本章将针对基于知识图谱的众测智能推荐技术进行需求分析，对于系统实现的前后端架构，推荐和重复率检测算法，实体类和数据库等关键部分进行概要设计和说明，对于系统的四个模块进行详细设计。

3.1　系统需求分析

3.1.1　涉众分析

本系统作为协作式众包测试平台，采用了基于知识图谱的众测智能推荐技术实现缺陷报告的推荐和重复性检测，系统的涉众分析如表3-1所示，主要涉众包括众测工人，评审专家，众测管理员。

众测工人作为众包测试任务的参与方，可以通过系统填写测试报告，测试用例，缺陷报告，查看报告详情，审核报告，筛选报告，优化报告等。他们不一定拥有很高的开发能力和测试能力，但他们来自不同地区，拥有不同设备，位于不同的测试环境，会采用不同的测试方法，他们希望系统能提供良好的用户体验，并能帮助他们进行协作，提高测试效率。

评审专家作为众包测试任务的评审方，可以通过系统对缺陷报告进行打分，决定缺陷报告的得分和优先级，间接影响最终交付报告的质量。评审专家

拥有一定的 Bug 定位和复现能力，希望系统提供查重能力，评审功能和良好的用户体验，提高评审效率。

众测管理员作为众包测试任务的管理方，可以通过系统创建众测任务，管理众包工人，查看缺陷报告，进行报告查重和报告整编，他们拥有一定的测试能力和管理能力，希望系统能够提供较好的查重服务和整编服务，提高最终交付报告的质量，满足任务发布方的需求。

表 3-1: 涉众分析表

涉众名称	涉众特征与期望
众测工人	作为众包测试任务的参与方，他们希望能在众包测试过程中获得系统的帮助，进行测试报告的提交，通过与他人进行协作减少重复劳动，发掘更多 Bug。众测工人应有一定的语言组织能力和协作能力，能够进行报告的填写和审核。应该具有一定的测试能力，能够进行 Bug 的发现，定位与复现。
众测管理员	作为众包测试任务的管理方，他们希望系统尽可能减少报告的重复率，提高测试页面覆盖率和报告质量。众测管理员应有一定的语言组织能力，进行缺陷报告的整编和交付。应该具有一定的管理能力，进行众测任务的众测工人的管理。
评审专家	作为众包测试任务的评审方，他们希望系统尽可能减少重复报告数，减少工作量，并提供良好的用户体验。评审专家应该具有一定的测试能力，能够定位 Bug 并进行复现，公平地对缺陷报告进行打分。

3.1.2 功能性需求

本系统主要用于众包工人进行众包测试，一场典型的众包测试流程如下：

1）任务发布方与众测网站进行合作，提供众包测试任务的相关信息，包括测试目标（网站或 APP），测试需求。

2）众测网站管理员根据测试目标和测试需求进行众包测试任务的创建，发布众测任务，通知众包工人。

3）众包工人进行报名，在众测任务规定时间内使用自己的测试方法进行测试，填写报告，包括测试报告，测试用例，缺陷报告，同时进行相互协作，包括报告审核，报告优化。

4）评审专家对缺陷报告进行评审打分。

5）众测管理员进行缺陷报告的整编聚合，剔除重复报告，整理一份完整的测试报告，发送给任务发布方，完成众包测试。

表 3-2: 系统功能需求表

ID	需求名称	需求描述
R1	填写测试报告	用户可以创建测试报告。用户进入众包测试界面，进行测试报告基本信息的填写，包括测试报告名称，测试设备品牌，测试设备名称，测试设备操作系统等。
R2	填写测试用例	用户可以创建测试用例。用户可以填写测试用例相关信息，包括用例名称，前置条件，测试步骤，预期结果等。
R3	填写缺陷报告	用户可以创建缺陷报告。用户可以填写缺陷报告相关信息，包括选择发现 Bug 位置，基本属性，所属用例，填写标题，描述，上传截图等。
R4	查看推荐报告	用户可以查看系统推荐的缺陷报告。用户在创建缺陷报告的过程中，系统会不断根据用户的输入推荐相似的缺陷报告，用户可以查看推荐报告列表，点击可以查看缺陷报告具体信息，包括缺陷报告 ID，所处位置，基本属性，所属用例，标题，描述，截图，父子关系等。
R5	筛选缺陷报告	用户可以通过筛选查看缺陷报告。用户在进行众测任务的过程中进入评审页面，可以通过三级页面进行缺陷报告的筛选，也可以在缺陷报告创建页面通过关键词搜索相关的缺陷报告，点击可以查看缺陷报告具体信息。
R6	审核缺陷报告	用户可以对缺陷报告进行审核。用户查看缺陷报告具体信息时，可以根据自己对该缺陷报告的评价，进行点赞点踩，点赞代表用户支持该缺陷报告，点踩代表用户反对该缺陷报告。
R7	克隆缺陷报告	用户可以对缺陷报告进行克隆。用户查看缺陷报告具体信息时，可以对该缺陷报告进行克隆，在此基础上进行修改和优化，进行提交，新报告作为被克隆报告的子报告进行保存。
R8	缺陷报告查重	众测管理员可以对缺陷报告进行重复性检测。众测管理员在众测任务完成后，可以根据要求对缺陷报告进行重复性检测，重复率可以自定义，检测出来重复的报告自动置零分，生成说明文档。

基于知识图谱的众测个智能推荐技术主要作用于步骤 3 和 4，在众包工人进行报告编辑的过程中，进行报告推荐，让用户进行审核和优化，提供筛选功能，利用群体智能从源头减少重复报告的产生。在众测任务结束后提供重复报告的检测服务，减少专家评审和报告整编时的重复报告数量，降低评审和整编的工作量，提高最终交付报告的质量。因此本系统的功能性需求列表如表3-2所示。

3.1.3 非功能性需求

性能：本系统应具有较高的性能。用户从操作到得到反馈，时间不超过200ms，部分重要操作如缺陷报告推荐应不超过 100ms。系统应能承受 2000 名用户并发使用并正常运行。

可用性：可用性是指系统在某个时间段内能够正常运行的可能性。本系统用于实际生产，应具有较高的可用性，在 99.9% 时间内能够正常运行。

可扩展性：可扩展性指系统适应不同业务环境和运行要求的能力。本系统应具有较高的可扩展性。系统应对主要模块和服务的接口，包括推荐模块，重复性检测模块进行抽象。系统应对机器学习相关算法的数据和程序进行分离。在新增系统模块和功能时，无需修改已有模块。在对已有模块进行重构时，不影响其他模块。

可维护性：系统的可维护性主要描述系统出现故障后定位和修复的能力，以及在现有基础上进行改进和优化的能力。本系统会进行长时间的实际应用，应有较高的可维护性。系统需要有完整的日志记录模块和详细的代码注释，并将各个模块进行抽象和解耦,减少运维人员的负担。

健壮性：健壮性主要衡量系统是否能在异常情况下进行工作。本系统的服务端和数据库服务器需要有一定的容灾能力。通过分布式部署，在单个服务器宕机时保证系统的可用。对数据库采用主从模式，出现故障时及时进行数据库切换，然后进行问题定位和修复，保证数据库的可用。

易用性：易用性指系统对用户来说易于学习和使用，能够减轻学习负担，提高用户的满意程度等。本系统的主要使用者是众包工人，应具有较强的易用性。系统通过学习文档和视频教学减轻用户的学习负担。系统前端界面简洁明了，文字说明丰富，反馈及时，人机交互体验好。用户应在十分钟内了解并熟悉系统的使用，可以开始进行众包测试。

3.1.4 系统用例描述

系统用例图如图3-1所示，将本系统分为八个用例，用户分为众测工人和众测管理员。其中有七个用例属于众测工人，分别为填写测试报告用例，填写测试用例用例，填写缺陷报告用例，查看推荐报告用例，筛选缺陷报告用例，审核缺陷报告用例，克隆缺陷报告用例；缺陷报告查重用例属于众测管理员。

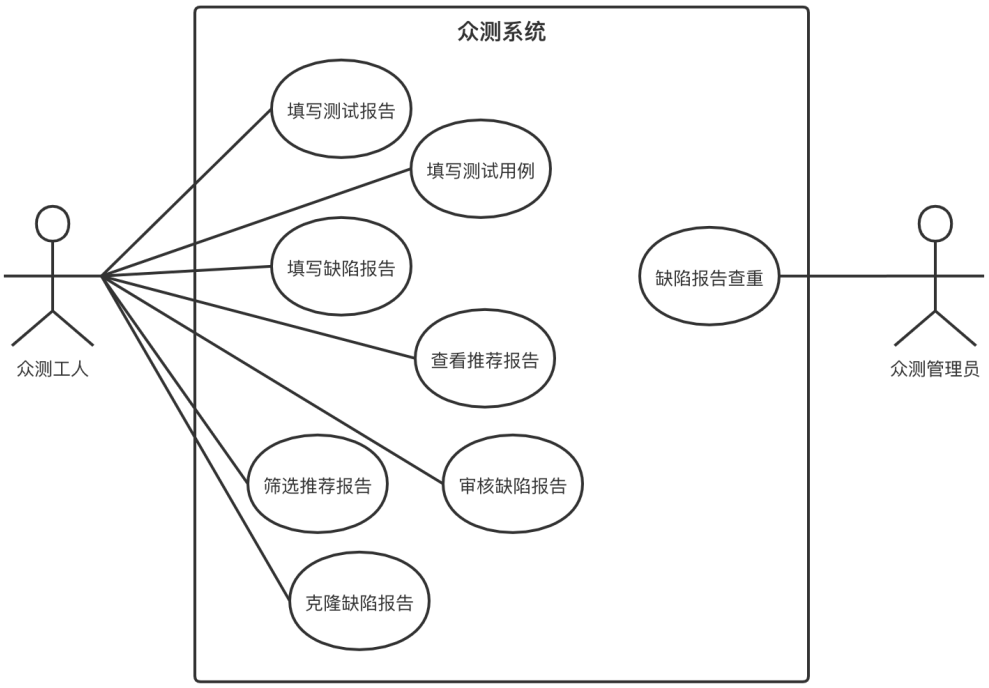

图 3-1: 系统用例图

填写测试报告是进行众测任务必经的一步。在众测任务开始后，众包工人点击众测网站中“编辑报告”按钮，可以进入众测报告填写页面。如果是首次进入，需要先进行测试报告相关信息的填写，才能进行其他操作。填写内容包括测试报告名称，测试设备品牌，测试设备名称，测试设备操作系统，这些信息方便任务发布方在同样的条件下进行缺陷的复现。完成后点击“保存”按钮，才可以进行测试用例和缺陷报告的创建。在众测任务进行的过程中，众包工人可以随时点击页面右上角的“编辑”按钮，进行测试报告基本信息的修改。众测工人编辑信息后若未保存，系统自动保存草稿。填写测试报告用例描述如表3-3所示。

表 3-3: 填写测试报告用例描述

UC1	填写测试报告
参与者	众测工人，目的是填写测试报告相关信息
触发条件	众测工人点击“编辑报告”按钮或“编辑”按钮
前置条件	众测工人已经过认证和授权
后置条件	系统保存测试报告相关信息
优先级	高
基本流程	1. 众测工人点击“编辑报告”按钮或“编辑”按钮； 2. 系统显示测试报告编辑页面； 3. 众测工人填写测试报告相关信息，包括测试报告名称，测试设备品牌，测试设备名称，测试设备操作系统，点击“保存”按钮； 4. 系统保存测试报告相关信息。
特殊需求	众测工人编辑信息后未保存，系统自动保存草稿。

表 3-4: 填写测试用例用例描述

UC2	填写测试用例
参与者	众测工人，目的是填写测试用例相关信息
触发条件	众测工人点击“创建用例”按钮
前置条件	众测工人已经过认证和授权
后置条件	系统保存测试用例相关信息
优先级	高
基本流程	1. 众测工人点击“创建用例”按钮； 2. 系统显示测试用例编辑页面； 3. 众测工人填写测试用例相关信息，包括测试用例名称，前置条件，测试步骤，预期结果，点击“保存”按钮； 4. 系统保存测试用例相关信息。
特殊需求	众测工人编辑信息后未保存，系统自动保存草稿。

填写测试用例是创建缺陷报告的必要条件，缺陷必然是在某一个测试用例下被发现的，一个测试用例过程中可能发现 0 到 N 个缺陷。众包工人点击“创

建用例”按钮，填写测试用例相关信息，包括测试用例名称，前置条件，测试步骤，预期结果。点击“保存”按钮，系统创建测试用例。众测工人编辑信息后若未保存，系统自动保存草稿。填写测试用例用例描述如表3-4所示。

填写缺陷报告是众包测试任务过程中最重要的部分。众包工人点击“创建Bug”按钮，进行缺陷报告的创建，包括选择页面，选择基本属性，填写标题和描述，上传截图几个步骤。

首先选择 Bug 所处页面：众测平台管理员在发布众包测试任务时，会将测试目标（移动应用或 Web 网页）进行梳理，根据测试目标的组织结构和逻辑，将之分解为三级页面的形式，一级页面代表用户进入测试目标后即可看到的页面，进行操作后才能进入下一级页面，一般测试目标的逻辑深度都不多于三级。以一次真实的众测任务“咕咚翻译众包测试”为例，该众测任务是针对咕咚翻译 APPDebug 版本进行的，共两个一级页面，七个二级页面，十九个三级页面，自顶向下覆盖了了咕咚翻译 APP 的所有使用页面，众包工人无论在哪里发现了 Bug，均可找到对应的页面进行选择。选择页面方便任务发布方快速定位 Bug 的位置，进行复现和排查。

然后，选择 Bug 的特征属性：本系统将描述 Bug 所需的基本属性划分为漏洞分类，严重等级和复现程度三种。其中漏洞分类属性代表众测工人认为该 Bug 所属的类别，可分为其他，安全，性能，页面布局缺陷，用户体验，功能不完整，不正常退出等七类。严重等级属性代表众测工人认为该 Bug 对于整个系统的危害程度，分为紧急，严重，一般，较轻，待定等五类。紧急一般代表阻塞性问题，造成系统崩溃，死机，死循环等，如网站无法打开，有病毒或脚本植入等；严重代表系统的实现与需求完全不符，重要模块功能丧失类问题，如系统主要功能无法运行，程序无故退出等；一般代表功能存在缺陷但不影响系统整体使用，如加载时间长，边界值错误，删除失败等。较轻代表不影响操作功能，但影响用户体验等可以优化的问题，如错别字，界面格式不规范等。复现程度属性代表在众测工人所使用的设备上复原 Bug 产生的难易程度，分为必现，大概率复现，小概率复现，无规律复现，其他等五类。最后，众测工人还需选择该 Bug 是在哪个测试用例下发现的，并选择该测试用例。根据基本属性，任务发布方可以对 Bug 的优先级进行排序，优先对更严重，出现频率更高的 Bug 进行处理。与测试用例相关联使得任务发布方能够用正确的步骤进行 Bug 的复现。

接着，填写 Bug 的标题和描述：众测工人需要用简单易懂，易于分类的描

述作为缺陷报告的标题，使用一两句话描述清楚出现 Bug 的页面，出现 Bug 的关键操作，Bug 的现象三个要点；在报告的描述中需要包含复现步骤和 Bug 现象两个要点，复现步骤要按自己的实际操作写清楚每一步，并按序号进行排序，便于任务发布方迅速了解 Bug 的实际情况，进行定位，复现和修复。

最后，上传 Bug 截图：除了使用文字外，截图也能帮助任务发布方理解 Bug 的真实情况，如所处页面，表现形式，严重程度等，众测工人可以上传该 Bug 的截图并进行标注，便于任务发布方迅速定位 Bug 位置，进行复现和处理。填写缺陷报告用例描述如表 3-5所示。

表 3-5: 填写缺陷报告用例描述

UC3	填写缺陷报告
参与者	众测工人，目的是创建缺陷报告
触发条件	众测工人点击“创建 Bug”按钮
前置条件	众测工人已经过认证和授权
后置条件	系统保存缺陷报告相关信息
优先级	高
基本流程	1. 众测工人点击“创建 Bug”按钮； 2. 系统显示缺陷报告编辑界面； 3. 众测工人填写缺陷报告相关信息，选择三级页面，漏洞分类，严重等级，复现程度和测试用例，填写标题和详细描述，上传 Bug 截图，点击“保存”按钮； 4. 系统保存缺陷报告相关信息。
特殊需求	众测工人编辑信息后未保存，系统自动保存草稿。

查看推荐报告是基于知识图谱的众测智能推荐技术的核心应用点。众包工人在创建缺陷报告的过程中，系统会不断根据用户的数据输入和交互行为推荐众包工人可能感兴趣的缺陷报告，众包工人被动接受报告推荐。众包工人可以查看，审核，克隆和优化缺陷报告。查看推荐报告可以让众包工人了解他人的进度和发现的 Bug，避免在已发现的 Bug 上浪费时间和精力，减少重复工作，从源头减少重复报告的产生。缺陷报告以列表的形式展现，粗略展示报告的标题，所处页面和推荐指数，点击后可以查看缺陷报告具体信息，包括缺陷报告 ID，所处位置，基本属性，所属用例，报告标题，具体描述，报告截图，父子关系等。查看推荐报告用例描述如表 3-6所示。

表 3-6: 查看推荐报告用例描述

UC4	查看推荐报告
参与者	众测工人，目的是查看系统推荐的缺陷报告
触发条件	众测工人编辑缺陷报告
前置条件	众测工人已经过认证和授权
后置条件	系统推荐相关缺陷报告
优先级	高
基本流程	1. 众测工人进行缺陷报告的编辑； 2. 系统根据众测工人的输入，向众测工人推荐缺陷报告，以列表的形式展现； 3. 众测工人点击感兴趣的缺陷报告； 4. 系统展示缺陷报告具体信息，包括缺陷报告 ID，所处页面，基本属性，所属用例，标题，描述，截图，父子关系等。

表 3-7: 筛选缺陷报告用例描述

UC5	筛选缺陷报告
参与者	众测工人，目的是根据条件或关键词筛选缺陷报告
触发条件	众测工人点击“所有 Bug”按钮
前置条件	众测工人已经过认证和授权
后置条件	系统展示筛选结果
优先级	中
基本流程	1. 众测工人点击“所有 Bug”按钮 2. 系统展示已提交缺陷报告页面 3. 众测工人选择缺陷报告的三级页面，特征属性等； 4. 系统根据特征条件筛选出缺陷报告，并以列表的形式展现；
扩展流程	1.1 众测工人输入关键词，如“登录”； 2.1 系统根据关键词筛选出缺陷报告，并以列表的形式展现；

筛选缺陷报告方便众包工人以主动的方式发现自己感兴趣的报告，提前查看是否已经有相似的报告提交，了解他人的提交情况，避免重复报告的产生。众包工人在进行众测任务的过程中点击“所有 Bug”按钮进入已提交缺陷报告页面，可以通过三级页面和其他基本属性进行缺陷报告的筛选，也可以在缺陷报告创建页面通过关键词搜索相关的缺陷报告，点击可以查看缺陷报告具体信息。筛选缺陷报告用例描述如表 3-7所示。

审核缺陷报告方便众包工人自发的对缺陷报告进行评价。众包工人查看缺陷报告具体信息时，可以根据自己对该缺陷报告的评价，点击“点赞”按钮进行点赞，点击“点踩”按钮进行点踩，点赞代表众包工人支持该缺陷报告，点踩代表众包工人反对该缺陷报告，众包工人的评价会影响缺陷报告的最终优先级，间接影响最终交付报告的质量。审核缺陷报告用例描述如表 3-8所示。

表 3-8: 审核缺陷报告用例描述

UC6	审核缺陷报告
参与者	众测工人，目的是审核缺陷报告，进行群体协作
触发条件	众测工人点击查看缺陷报告具体信息
前置条件	众测工人已经过认证和授权
后置条件	系统保存审核结果
优先级	中
基本流程	1. 众测工人点击查看缺陷报告具体信息； 2. 系统展示缺陷报告具体信息； 3. 众测工人点击“点赞”按钮进行点赞或点击“点踩”按钮进行点踩； 4. 系统保存审核结果。
扩展流程	3.1 众测工人再次点击，取消审核行为； 4.1 系统删除众测工人对该缺陷报告的审核结果。

克隆缺陷报告利用群体智能提高众包工人创建报告的效率。众包工人查看缺陷报告具体信息时，可以点击“克隆”按钮进行缺陷报告的克隆，复制该缺陷报告的内容至缺陷报告填写界面，在此基础上进行修改和优化，除三级页面外其余信息均可修改，编辑结束后点击“保存”按钮，新创建的缺陷报告作为被克隆的缺陷报告的子报告进行提交。克隆缺陷报告用例描述如表 3-9所示。

表 3-9: 克隆缺陷报告用例描述

UC7	克隆缺陷报告
参与者	众测工人，目的是克隆他人的缺陷报告
触发条件	众测工人点击“克隆”按钮
前置条件	众测工人已经过认证和授权
后置条件	系统帮助众测工人进行缺陷报告克隆
优先级	中
基本流程	1. 众测工人点击查看缺陷报告具体信息； 2. 系统展示缺陷报告具体信息； 3. 众测工人点击“克隆”按钮； 4. 系统展示缺陷报告编辑页面，将被克隆报告的相关信息填入； 5. 众测工人在被克隆报告的基础上进行编辑，点击“保存”按钮； 6. 系统保存缺陷报告相关信息。

表 3-10: 缺陷报告查重用例描述

UC8	缺陷报告查重
参与者	众测管理员，目的是对众测工人提交的缺陷报告进行重复性检测
触发条件	众测管理员点击“重复性检测”按钮
前置条件	众测管理员已经过认证和授权
后置条件	系统进行缺陷报告重复性检测
优先级	高
基本流程	1. 众测管理员输入自定义的重复率，点击“重复性检测”按钮； 2. 系统根据重复率进行查重，展示重复报告相关信息； 3. 众测管理员进行人工复核，取消不重复报告的“是否置零”勾选，点击“报告置零”按钮； 4. 系统将重复报告的参考分置为零分，产生说明文档，可供下载。

缺陷报告查重减少评审专家和报告整编的工作量。众测管理员在众测任务完成后，根据需要自定义重复率，输入重复率后，点击“重复性检测”按钮，对缺陷报告进行重复性检测，众测管理员对系统认为重复的报告进行人工复核，对于不重复的报告，取消分数置零的选项，避免自动化查重出错，带来争议。复核结束后点击“报告置零”按钮，将参考分置为零分。系统产生说明文档，众测管理员可以下载，便于复盘和对众测工人进行说明。缺陷报告查重用例描述如表3-10所示。

3.2 系统概要设计

3.2.1 系统架构设计

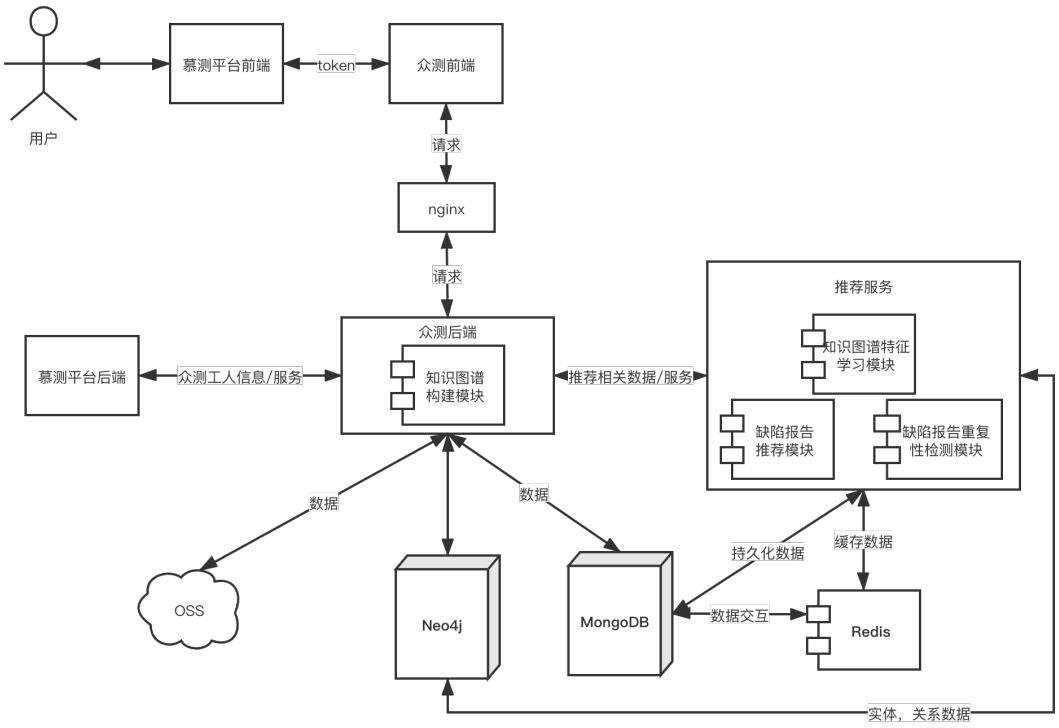

图 3-2: 系统架构设计图

本系统为基于测试平台开发的众包测试平台，采用前后端分离的开发方式。前端使用 Angular2 开发单页应用，众测后端服务使用 SpringBoot 开发，以微服务的形式向前端提供 Restful 服务，同时需要与测试平台进行通信，创建

众包测试任务，获取众测工人信息等。考虑到基于知识图谱的众测智能推荐技术需要使用机器学习相关技术，python 语言对机器学习的支持较好，因此使用 Django 进行推荐和查重服务的开发，并向众测后端提供推荐和查重相关服务。具体设计如图 3-2所示，图中各服务节点均打包成为 Docker 容器，采用 Jenkins 自动化部署运维，提高部署效率。

1）以 Angular2 框架进行前端开发，采用单页面开发模式，以 TypeScript 作为业务逻辑的开发语言，使用 Echarts 进行图表的构建和展示，使用开源工具库 Bootstrap 提高开发效率，采用 npm 对前端包进行管理和打包，将前端项目打包为静态文件，部署上线。

2）以 SpringBoot 框架进行后端主体开发，采用经典的 MVC 模式进行代码开发，Controller 层进行请求的接受和解析，并转发至对应的 Service 层进行处理；Service 层进行请求的逻辑处理，与外部平台和推荐服务进行通信等，如有对于数据的操作，将之转发至 Dao 层；Dao 层进行数据库相关处理，如数据存储，更新，删除，查询等。

3）以 Django 框架进行推荐服务的开发，采用基于知识图谱的众测智能推荐技术，在众测任务进行过程中为众包测试工人提供缺陷报告推荐的服务，在众测任务结束后向众测管理员提供缺陷报告重复性检测的服务。推荐服务以微服务的形式向众测平台后端提供服务，众测平台 Service 层访问推荐服务提供的 Restful 接口并获取服务。

4）以 MongoDB 作为后端和推荐服务的持久化数据库，进行文档化数据的存储；以 Redis 作为推荐服务的缓存数据库，减少对数据库的访问，降低数据库的并发压力，提高推荐服务的效率；以阿里云 OSS 作为文件和图像的存储，提高大数据文件的存取能力；以 Neo4j 作为图数据库进行知识图谱相关数据的存储，提供完善的图查询语言，提高查询速度。

3.2.2 系统模块划分

如图 3-2所示，本系统划分为四个模块，分别为知识图谱构建模块，主要是在众包工人进行众测任务，创建缺陷报告的过程中，利用众包工人本身的信息和提交的测试报告数据，实时的进行该众测任务相关的知识图谱的构建；知识图谱特征学习模块，针对众包工人提交的数据进行关键词，实体和关系的特征提取；缺陷报告推荐模块，在众测任务进行过程中进行缺陷报告的实时推荐；缺陷报告重复性检测模块，在众测任务完成后对提交的缺陷报告进行查

重。其中，知识图谱构建模块主要位于众测后端服务中，其余三个模块位于推荐服务中。下面根据一次完整的众测任务进行模块的讲解。

知识图谱构建模块：众测任务进行过程中，众包工人进行测试报告，测试用例，缺陷报告的创建，系统将文档化数据保存至 MongoDB，将图像和文件数据保存至 OSS，同时将数据以实体和关系的形式保存至 Neo4j 数据库，进行知识图谱的构建，用于后续的特征学习和报告推荐，查重。

知识图谱特征学习模块：系统利用历史关键词数据进行词向量模型的训练，利用历史实体和关系数据进行知识图谱特征模型的训练，众测过程中系统收集到工人提交的缺陷报告的行为信息，利用词向量模型获得关键词的特征，利用知识图谱特征学习模型获得实体和关系的特征，用于缺陷报告推荐模块和重复性检测模块的使用。

缺陷报告推荐模块：众包工人在创建缺陷报告的过程中，系统会不断根据众包工人输入的数据，采用基于知识图谱的众测智能推荐技术，推荐众包工人可能会感兴趣的缺陷报告，众包工人也可以主动进行筛选查找感兴趣的报告。众包工人可以查看报告内容，了解他人进度和成果，减少重复工作；众包工人可以对这些报告进行审核，工人的意见最终会影响报告的优先级和最终报告的质量；众包工人也可以对报告进行克隆和优化，利用群体智能提高报告描述质量，提高众包测试的效率。

缺陷报告重复性检测模块：众测任务完成后，会请评审专家针对众包工人提交的缺陷报告进行审核打分，确定缺陷报告的优先级。众测管理员可以进行缺陷报告重复性检测，系统会根据管理员提交的重复率要求，进行查重，众测管理员进行人工复核后将重复报告的参考分置为零分，减少评审专家和报告整编的工作量，同时生成说明文档，用于复盘和对众包工人进行说明。

3.2.3 系统视图模型

软件架构用来描述软件结构的设计，但单一视图难以完整进行描述，本小节参考 PB Kruchten[55] 提出的“4+1”视图，给出系统的架构设计，并从场景视图，开发视图，逻辑视图，进程视图和物理视图五个方面进行详细描述。场景视图已在本文的系统用例图部分进行了介绍，此处不再赘述，接下来从其他四个视图的角度对系统进行说明。

1）逻辑视图：逻辑视图是从用户服务的角度描述系统架构，对系统进行逻辑分层，模块划分，依赖关系分析，可以用 UML 中的类图进行描述。本系

统的逻辑视图如图 3-3所示，将系统模块进行抽象进行逻辑视图构建。

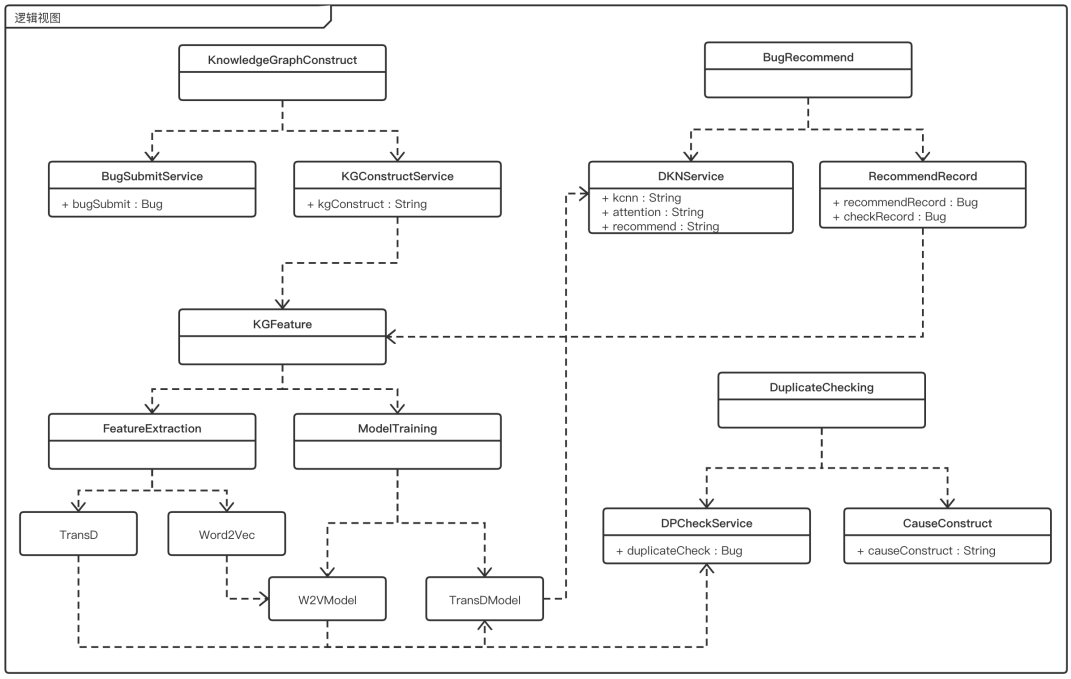

图 3-3: 逻辑视图

KnowledgeGraphConstruct 模块进行知识图谱的构建，BugSumbitService 负责测试报告具体内容的填写上传，将数据存储至 MongoDB 和 OSS；KGConstructService 负责知识图谱的构建，包括知识抽取，实体识别，关键词提取，关系定义等，将数据存储至 Neo4j 数据库。KGFeature 模块使用历史知识图谱数据进行模型训练，使用模型对实时上传的数据进行特征提取，其中 ModelTraining 负责模型训练和学习，使用关键词数据进行 Word2Vec 模型的训练，使用知识图谱实体和关系数据进行 TransD 模型的训练；FeatureExtraction 负责特征提取，使用 Word2Vec 模型提取词语特征，使用 TransD 模型提取实体和关系的特征。BugRecommend 模块向用户进行缺陷报告推荐，其中 DKNService 使用 KCNN 模型获取缺陷报告的特征，使用基于注意力机制的用户兴趣预测网络计算众包工人点击某份缺陷报告的概率；RecommendRecord 对推荐记录和审核记录进行存储，反馈给 KGFeature，用于优化模型。DuplicateChecking 模块进行缺陷报告重复性检测，其中 DPCheckService 采用 Word2VecModel 获取词向量，采用基于 SIF 的加权算法获取句子向量，采用余弦距离计算缺陷报告相似度；CauseConstrcut 进行说明文档的生成和下载。

2）开发视图：开发视图从软件开发人员的角度描述系统的模块组织与管理。如图 3-4所示，系统开发主要分为 UI 和 Service Logic 两部分。

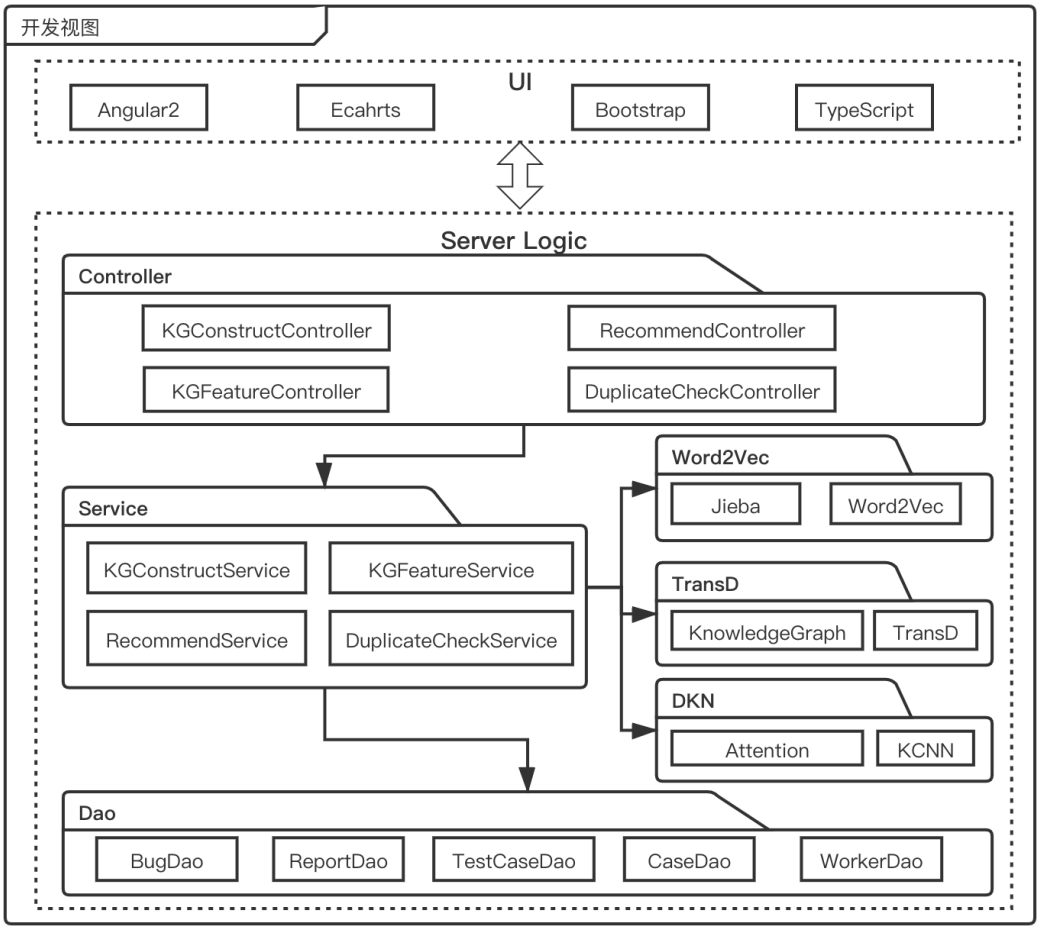

图 3-4: 开发视图

UI 部分主要使用 Angular2 作为开发框架进行单页应用开发，以 TypeScript 作为业务逻辑主要的开发语言，使用 ECharts 进行前端图表的构建和展示，结合 Bootstrap 中的开源工具包提高开发速度和效率。

Service Logic 部分按照功能和模块进行划分。Controller 层进行请求的接受和转发，KGConstructController 提供知识图谱构建相关接口，KGFeatureController 提供知识图谱特征学习相关接口，RecommendController 提供缺陷报告推荐服务相关接口，DuplicateCheckController 提供缺陷报告重复性检测相关接口。Service 层进行实际的业务逻辑处理，实现 Controller 向外提供的接口，主要通过调用子模块实现实际功能。Word2Vec 模块进行 Word2Vec 模型的学习和

训练，获取关键词文字特征，用于缺陷报告推荐和重复性检测；TransD 模块进行 TransD 模型的学习和训练，获取实体和关系特征，用于缺陷报告推荐；DKN 模块进行 DKN 模型的学习与训练，对缺陷报告的特征提取，使用基于注意力机制的用户兴趣预测，得到用户点击某份缺陷报告的概率，实现缺陷报告推荐。Dao 层负责实体类和数据库相关操作，对 RedisAPI，MongoDBAPI，Neo4jAPI 和 OSSAPI 进行封装，与数据库及云服务进行交互。

3）进程视图：进程视图从系统处理请求的角度描述系统的并发请求，线程之间的交互与通信过程，重视系统的并发性。本系统的进程视图如图 3-5所示。

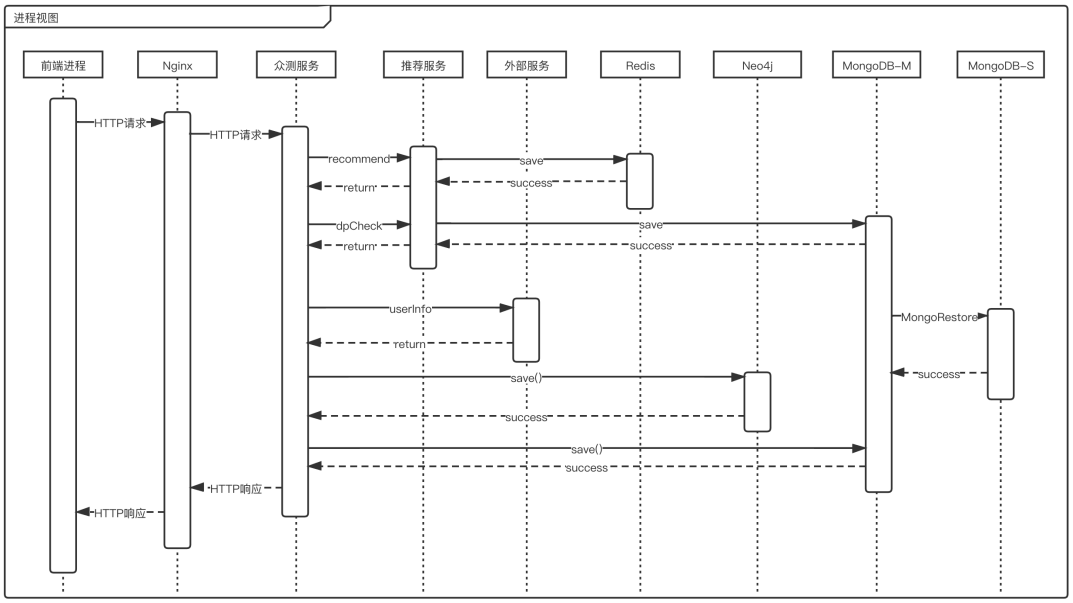

图 3-5: 进程视图

用户在众测前端进行操作，触发前端方法，发送 HTTP 请求至 Nginx 服务器，Nginx 根据配置进行请求的解析，根据负载均衡算法将请求发送至对应的服务端进程。众测后端采用分层模式，Controller，Service，Dao 层层调用，进行请求的处理。如果是推荐和查重相关的请求，众测服务端会采用 RESTful 的方式请求推荐服务进行处理。若众测服务端需要外部数据，如众测工人数据等，请求外部测试平台服务进行处理。若涉及到对数据的增删改查，服务端会与 MongoDB 和 Redis 进行交互。对于持久化数据，服务端进程会与 MongoDB 进行交互，将数据持久化至数据库主库，数据库会自动进行数据的备份。同时会对数据进行实体抽取，关系定义，获取知识图谱实体和关系，与 Neo4j 进

行交互，将知识图谱数据存储至图数据库。对于热点数据，服务端进程会与 Redis 进行交互，如果不存在，则先去 MongoDB 进行查询，得到结果后将该数据存储至 Redis，下次请求就可以直接从 Redis 中获得结果，不需要访问数据库，以此提高数据处理的效率，提高系统的并发性。服务端将业务请求处理完成后，会将响应数据返回给 Nginx 服务器，Nginx 将数据返回给前端，前端对数据进行解析并按照页面逻辑进行展示。

4）物理视图：物理视图是从运维人员的角度描述系统安装，部署，物理节点之间的通信问题。本系统的物理视图如图 3-6所示。

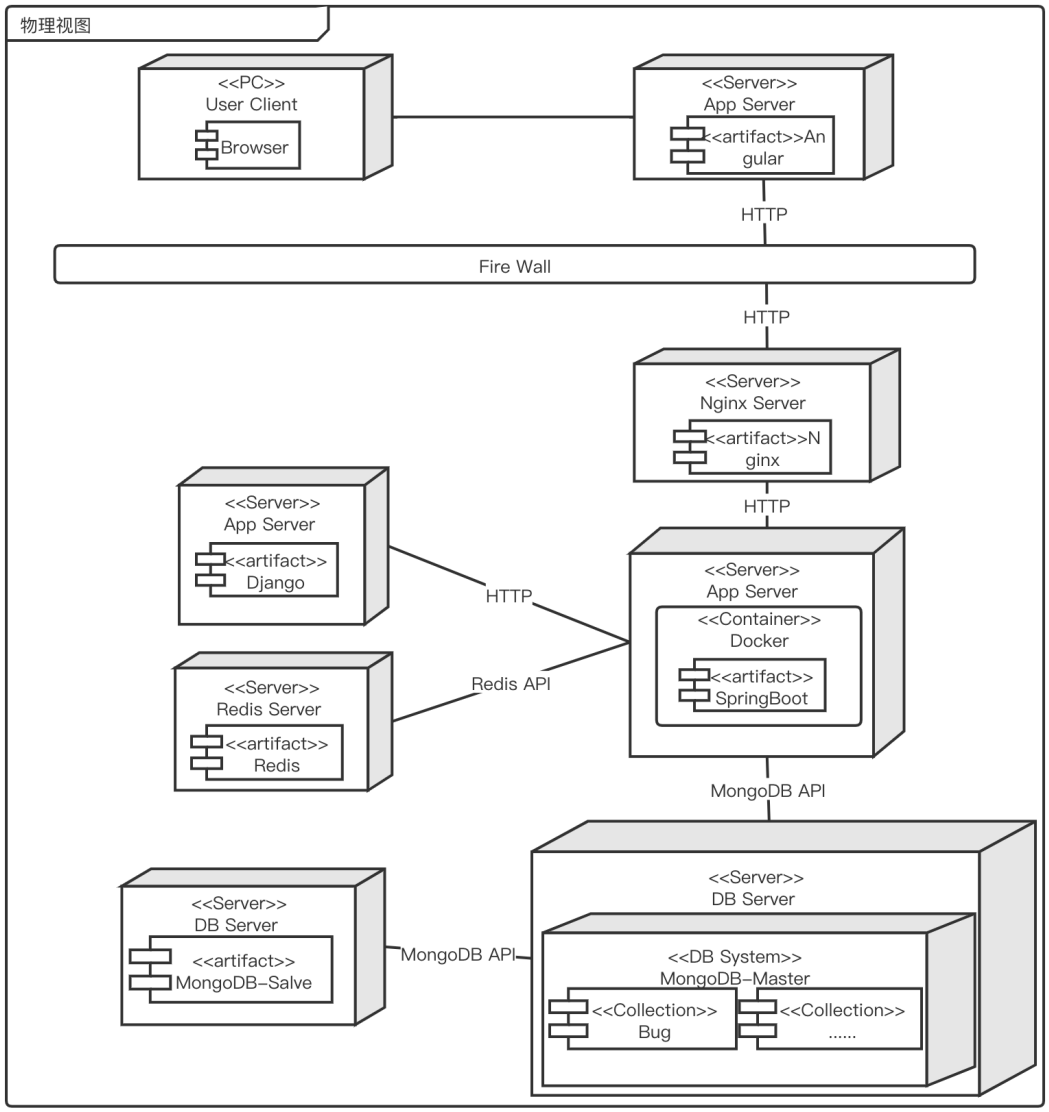

图 3-6: 物理视图

用户使用自己的计算机，通过浏览器访问测试平台主站前端网页，经过验证和授权获得权限。跳转至众测前端网页，众测前端以静态文件的形式部署在阿里云服务器上。通过众测前端向众测服务端发送 HTTP 请求，首先通过防火墙过滤掉不安全的请求。随后请求发送至 Nginx 服务器，Nginx 服务器进行解析后，采用负载均衡将请求转发至多个众测服务端。考虑到机器成本和运维能力，众测服务端，推荐服务，缓存数据库 Redis，数据库 MongoDB 均以 Docker 容器的形式单独部署于阿里云服务器上，以微服务的方式对外提供服务。服务端之间采用 RESTful API 的形式进行通信，服务端通过 API 与 MongoDB 和 Redis 进行交互。MongoDB 采用主从模式，主数据库主要接受写请求，从数据库主要接受读请求，定期与主数据库进行数据同步，主从模式提高了系统的可用性和可靠性。

3.2.4 实体类设计

基于知识图谱的众测智能推荐技术中，重要的实体类包括测试目标类 Case，众测工人类 Worker，测试报告类 Report，测试用例类 TestCase，缺陷报告类 Bug 等，基本实体类的设计及联系如图 3-7所示，一个工人可以参与 0 到 N 场众包测试任务，一场众测任务可以有 0 到 N 个工人参加。在一场众测任务中，一个工人创建一份测试报告，其中包含多个测试用例和多份缺陷报告，一个测试用例下可以发现 0 到 N 个缺陷，创建 0 到 N 份缺陷报告。

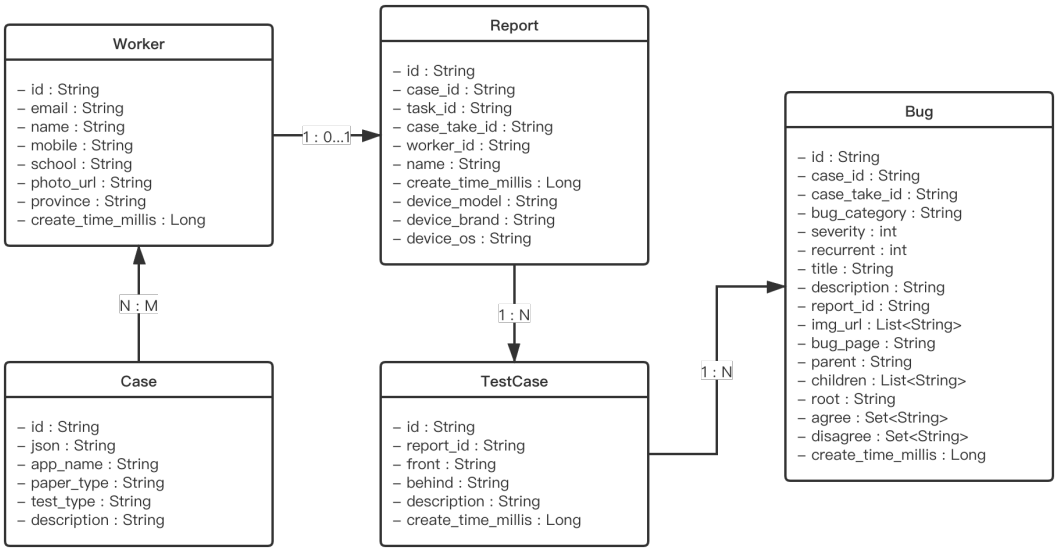

图 3-7: 实体类设计图

Case 类：Case 类存储众包测试中测试目标的相关信息。任务发布方与众测网站达成合作后上传测试目标（移动应用或 Web 网页），测试需求，奖励方式等必要信息，由网站管理员进行众包测试任务的创建和发布，Case 类具体内容如表 3-11所示：

表 3-11: Case 类设计描述

属性	数据类型	含义
id	String	测试目标 ID
json	String	测试目标三级页面 JSON 格式
app_name	String	测试目标名称
paper_type	String	测试目标所需填写的测试报告格式配置项
test_type	String	测试目标类型
description	String	测试目标描述

Report 类：Report 类存储测试报告的相关信息。测试报告是测试用例和缺陷报告的载体。众包测试过程中，众测工人需要进行测试报告的填写，主要内容是工人使用的测试设备相关信息，便于任务发布方使用相同设备进行缺陷的复现和排查。Report 类具体内容如表 3-12所示：

表 3-12: Report 类设计描述

属性	数据类型	含义
id	String	测试报告 ID
case_id	String	测试目标 ID
task_id	String	练习或考试或比赛 ID
case_take_id	String	众测任务 ID
worker_id	String	众测工人 ID
name	String	测试报告名称
create_time_millis	Long	测试报告创建时间
device_model	String	测试设备名称
device_brand	String	测试设备品牌
device_os	String	测试设备操作系统

TestCase 类：TestCase 类存储测试用例的相关信息。众包测试过程中，众测工人需要进行测试用例的填写，便于任务发布方进行了解测试覆盖度，进行 bug 的重现，TestCase 类具体内容如表 3-13所示：

表 3-13: TestCase 类设计描述

属性	数据类型	含义
id	String	测试用例 ID
report_id	String	测试报告 ID
name	String	测试用例名称
front	String	测试用例前置条件
behind	String	测试用例后置条件
description	String	测试用例详细描述
create_time_millis	Long	测试用例创建时间

Worker 类：Worker 类存储众测工人相关信息，主要来自测试平台主站，用于信息展示和聚合，具体内容如表 3-14所示：

表 3-14: Worker 类设计描述

属性	数据类型	含义
id	String	众测工人 ID
email	String	邮箱地址
name	String	众测工人昵称
mobile	String	手机号码
school	String	所属学校
photo_url	String	头像 url 地址
province	String	所属省份
create_time_millis	Long	众测工人账号创建时间

Bug 类：Bug 类存储缺陷报告的相关信息。众包测试过程中，众测工人需要进行缺陷报告的填写，以文字和截图的方式描述缺陷的所处页面，漏洞种类，严重程度，复现程度，表现形式，并与测试用例进行关联，便于任务发布方进行 Bug 定位和复现，提高修复 Bug 的效率。同时，Bug 类还存储了 Bug 之

间的亲缘关系和工人的审核记录，用于报告审核时的页面展现和参考，Bug 类具体内容如表3-15所示：

表 3-15: Bug 类设计描述

属性	数据类型	含义
id	String	测试用例 ID
case_take_id	String	众测任务 ID
bug_category	String	Bug 漏洞类型
severity	int	Bug 严重程度
recurrent	int	Bug 复现程度
report_id	String	测试报告 ID
title	String	缺陷报告标题
description	String	缺陷报告详细描述
img_url	List<String>	缺陷报告截图地址集合
bug_page	String	Bug 所属三级页面
case_id	String	测试目标 ID
parent	String	父缺陷报告 ID
children	List<String>	子缺陷报告 ID 集合
root	String	所属缺陷报告树的根结点 ID
agree	Set<String>	赞同的测试报告集合
disagree	Set<String>	反对的测试报告集合
create_time_millis	Long	缺陷报告创建时间

3.2.5 数据库设计

由于本系统主要存储的数据是众测工人填写的缺陷报告这种文档化数据，基于 MongoDB 面向集合，易于进行对象存储，支持类似 JSON 的 BSON 数据结构的特点，选用非关系型数据库 MongoDB 进行数据的存储。本系统为实体类设计中的每一个实体类创建一个集合 Collection，类似于关系型数据库中的表，每一行数据为文档 Document，多个文档组成一个集合，多个集合逻辑上组织在一起，就是数据库。数据库的 ER 图如图3-8所示。

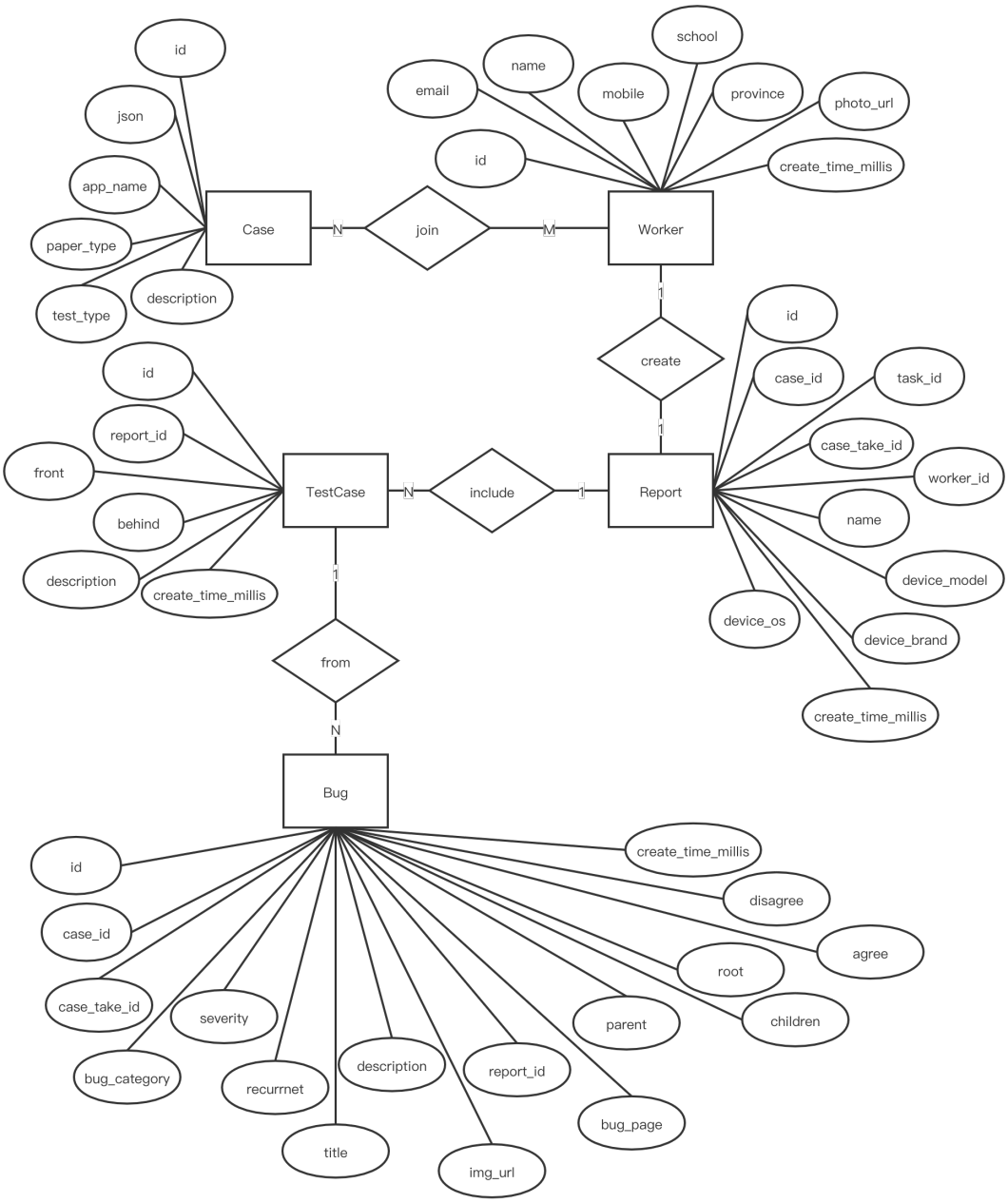

图 3-8: 数据库 ER 图

所有实体类对应的 Collection 属性与实体类设计字段类型，字段属性，字段含义一一对应，仅因为数据库字段类型限制进行一定的修改进行适配，此处仅以 MongoDB 中 Bug Collection 对应实体类 Bug 进行介绍，具体内容如表 3-16所示，其余 Collection 不做赘述。

表 3-16: Bug Collection 文档描述

属性	数据类型	含义
id	String	测试用例 ID
case_take_id	String	众测任务 ID
bug_category	String	Bug 漏洞类型
severity	int32	Bug 严重程度
recurrent	int32	Bug 复现程度
report_id	String	测试报告 ID
title	String	缺陷报告标题
description	String	缺陷报告详细描述
img_url	Array	缺陷报告截图地址集合
bug_page	String	Bug 所属三级页面
case_id	String	测试目标 ID
parent	String	父缺陷报告 ID
children	Array	子缺陷报告 ID 集合
root	String	所属缺陷报告树的根结点 ID
agree	Array	赞同的测试报告集合
disagree	Array	反对的测试报告集合
create_time_millis	String	缺陷报告创建时间

3.3 基于知识图谱的众测智能推荐技术

针对目前众包测试平台所面临的问题，本文从众包测试平台的角度出发，提出了基于知识图谱的众测智能推荐技术，提供了众包测试过程中的推荐服务和众包测试结束后的重复性检测服务。

推荐算法主要是在众测任务进行过程中，向众测工人推荐可能感兴趣的缺陷报告，让众测工人了解他人的进度，减少重复报告的产生，提高总体效率。众包工人可以对报告进行审核，影响缺陷报告优先级和可信度的排序；通过对缺陷报告进行克隆优化，可以利用群体智能提高缺陷报告的质量。由于缺陷报告推荐具有稀疏性和冷启动的问题，因此使用知识图谱作为辅助信息，丰富系统对于众包工人和缺陷报告的特征描述，增强推荐算法的挖掘能力。

重复率检测算法主要是在众测任务结束后，众测管理员对于众包工人提交的缺陷报告进行重复性检测，发现重复报告，在众测管理员人工复核后，将参考分置为零分，减少评审专家的工作量，提高最终交付报告的质量。

基于知识图谱的众测智能推荐技术通过及时的反馈和信息交流提高了众包工人的积极性，减少了重复报告的产生，提高了报告质量和测试页面覆盖率，满足了企业和个人的业务需求。

3.3.1 推荐算法

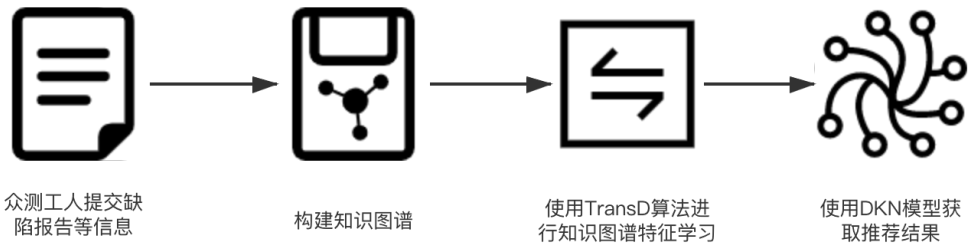

图 3-9: 推荐算法流程图

推荐算法具体流程如图3-9所示。首先，众包工人进行众测任务，提交缺陷报告等信息；系统获取众包工人提交的数据，结合收集到的众测工人个人信息和行为信息进行知识图谱的构建，这两步主要由知识图谱构建模块负责；然后使用 TransD 模型进行知识图谱特征学习，获取众测任务中的实体和关系向量，使用 Word2vec 模型进行词向量特征学习，获取缺陷报告的关键词向量，主要由知识图谱特征学习模块负责；最后使用 DKN 模型，根据用户历史关注过的报告，获取用户点击某份缺陷报告的概率，向用户推荐点击概率较大的报告，由缺陷报告推荐模块负责。

首先进行知识图谱的构建，数据来源主要包括众测工人的个人信息，提交的测试报告，测试用例，缺陷报告，众测工人的行为信息等。系统从这些数据中进行信息抽取，找出关键实体，定义出它们之间的关系，将实体和关系存储至 Neo4j 数据库，并与历史数据进行连接，进行知识图谱的构建，随着众测工人参加的众测任务越来越多，知识图谱的信息会越来越多，用于推荐和重复性检测的辅助信息也更加丰富。

表 3-17: 知识图谱实体关系表

实体 1	关系	实体 2
Worker	join	Case
Worker	create	Report
Case	includeR	Report
Report	includeT	TestCase
TestCase	includeB	Bug
Bug	includeK	KeyWord
Report	fork	Report
Worker	good	Bug
Worker	bad	Bug

系统主要实体和它们之间的关系如表3-17所示，由于众包测试系统的逻辑组织形式较为清晰，数据可以用结构化的方式进行表示，因此本文采用自顶向下的模式进行知识图谱的构建，先对实体，关系和属性进行定义，然后将数据进行存储。本系统涉及到的重要实体包括测试目标 Case，测试报告 Report，测试用例 TestCase，缺陷报告 Bug，众测工人 Worker，除此之外，还有关键词对应的实体 KeyWord。

与其他实体不同，KeyWord 实体与缺陷报告的关键词对应，首先对缺陷报告使用开源工具 jieba 进行分词，使用停用词表进行关键词筛选，然后进行同义词替换，将多个同义词进行融合，定义为同一个实体，如关键词“按钮”，“按键”均指向同一个实体“按钮”。

构建过程中并不需要将所有的数据都存入图数据库，大部分数据仍存入 MongoDB，只有与推荐相关的实体和属性才会参与知识图谱的构建，保证知识图谱的轻量级和专业性。本文构建的知识图谱以三元组进行表示，用实体-关系-实体的形式表示两个实体之间的关系，比如 Report 与 TestCase 就是 includeT 关系，一份测试报告中包含 0 到多个测试用例，Worker 与 Case 就是 join 关系，一个众测工人可以参加多次众测任务，一场众测任务也有多位众测工人参加。

实际构建出知识图谱简要示意图如图3-10所示，具体真实知识图谱可见第四章实现部分。

知识图谱构建完成后，需要进行知识图谱特征学习，学习实体和关系的特征，本文选用基于距离的翻译模型中的 TransD 算法。TransD 模型通过动态映

射矩阵获取知识图谱实体特征，并以向量形式进行表示，不仅考虑了关系的多样性，也考虑了实体的多样性，同时通过较少的参数降低了训练的时间复杂度，适用于大型知识图谱的训练和学习。由于本系统的历史数据较多，大概有上百场众测任务，上万条缺陷报告记录，构建的知识图谱数据量较大，因此选用该模型进行知识图谱实体特征的学习。

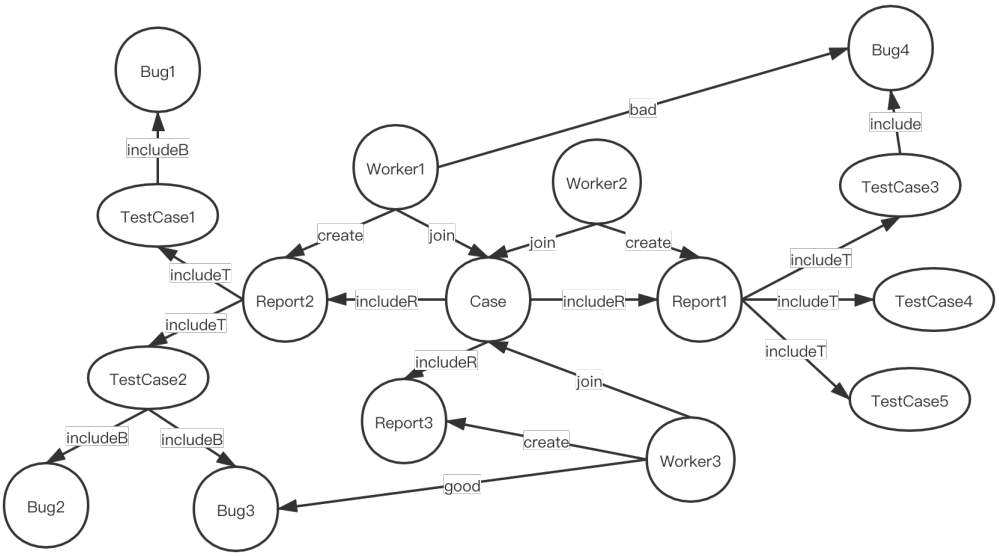

图 3-10: 知识图谱实体关系图

如图3-11所示，左边为实体空间，右边为关系空间，在关系空间中实现头尾实体与关系的投影，给头尾实体定义了不同的投影矩阵，避免因头尾实体差异过大导致误差，且投影矩阵反映了实体与关系的交互过程，与三者都相关。

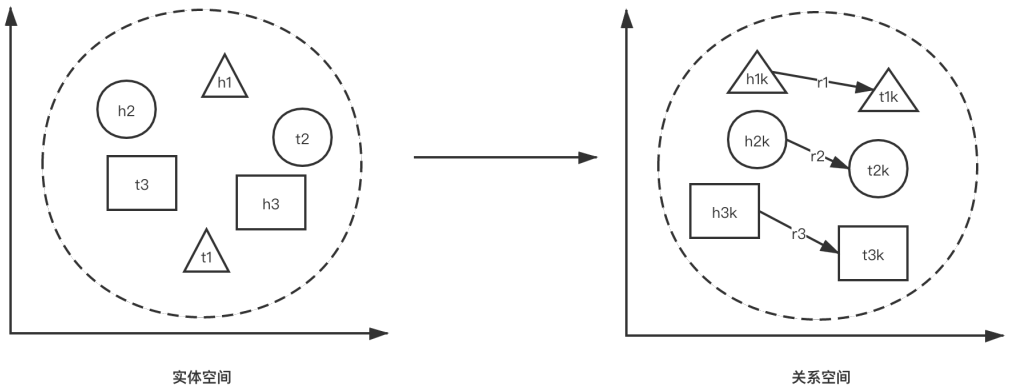

图 3-11: TransD 模型图

如公式3-1,3-2所示，给定一个三元组 (h,r,t)，TransD 将头实体和尾实体分别投影到关系空间，得到投影矩阵 M_{rh} 和 M_{rt}，I 为单位矩阵。

$$\mathbf{M}_{rh} = \mathbf{r}_p \mathbf{h}_p^\top + \mathbf{I}^{m \times n} \tag{3-1}$$

$$\mathbf{M}_{rt} = \mathbf{r}_p \mathbf{t}_p^\top + \mathbf{I}^{m \times n} \tag{3-2}$$

如公式3-3所示,获取投影矩阵之后，便可以计算头尾实体的投影向量。

$$\mathbf{h}_\perp = \mathbf{M}_{rh}\mathbf{h}, \quad \mathbf{t}_\perp = \mathbf{M}_{rt}\mathbf{t} \tag{3-3}$$

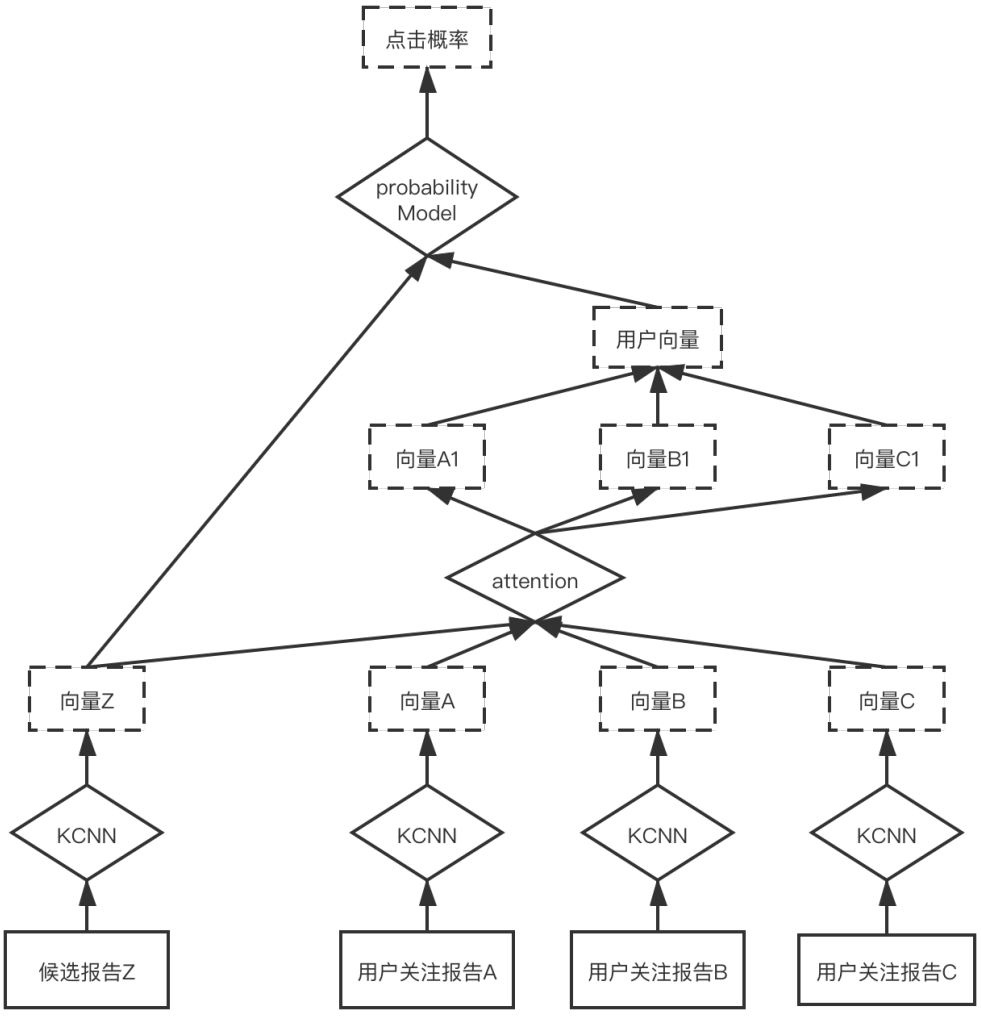

图 3-12: DKN 模型图

最后，采用依次学习 DKN 模型将知识图谱特征学习与推荐系统相结合。如图3-12所示，DKN 模型主要包括两个部分，第一部分是基于卷积神经网络的文本特征提取机制 KCNN，输入候选报告集合，用户历史关注的报告列表，通过 KCNN 来提取报告的特征。第二部分是 Attention Net，使用注意力机制计算两种向量之间的权重，得到用户向量表示，最后得到用户对此报告的点击概率。DKN 模型的训练输入数据包括历史缺陷报告关键词向量，历史缺陷报告实体向量，候选缺陷报告关键词向量，候选缺陷报告实体向量和标签 label，说明用户是否关注此历史报告。

KCNN 全称 Knowledge-aware CNN，是基于卷积神经网络的文本特征提取方法，输入包括文本的关键词向量、词语对应的实体向量和实体上下文向量，用句子所包含词的词向量组成的二维矩阵，经过一层卷积操作之后再做一次 max-over-time 的池化操作得到缺陷报告的最终向量表示。KCNN 能够保留词语和相关实体之间的联系和对齐方式，并在不同的向量空间中进行卷积，同时考虑到词语向量和实体向量的最佳维度可能不同，因此先将实体向量和实体上下文向量映射到同一个空间，再将三部分向量进行拼接，作为 KCNN 的输入。

文本词向量通过 Word2vec 模型获得，而实体向量可以通过 TransD 模型获得。一个实体 e 的上下文实体由 e 的所有邻居节点组成，因此 e 的上下文实体向量则为 e 的所有邻居实体的特征的平均值。如公式3-4所示，e_i 为实体 e 的邻居实体，$context(e)$ 为邻居实体组成的集合，计算算数平均值即可。

$$\bar{\mathrm{e}}=\frac{1}{|\ \mathrm{context}\ (e)\ |}\sum_{e_i\in\ \mathrm{context}\ (e)}\mathbf{e}_i \tag{3-4}$$

在候选报告和用户关注报告经过 KCNN 得到最终向量表示后，采用基于注意力机制的用户兴趣预测，计算候选报告对于用户每份关注报告的 attention，再做加权求和，获得用户的向量表示。Attention 机制的参数少，复杂度低，对算力的要求小；可以和 CNN 一样进行并行处理，解决了 RNN 不能并行计算的问题；同时不会丢失长距离的重要信息，效果更好。

最终获得用户对于一份缺陷报告的点击概率，由此可以将用户更愿意点击的缺陷报告进行推荐，提高推荐的准确率和效率。

3.3.2 重复率检测算法

Word2Vec 计算文字相似度的流程如图3-13所示。众包工人提交缺陷报告后，系统首先对于文本描述进行分词，本文选用开源工具 jieba 分词，速度快，效果好；然后利用停用词表去除停用词，进行关键词的筛选和存储，关键词获取由知识图谱构建模块负责；接着使用预训练好的 Word2Vec 模型获取词语对应的词向量，模型训练由知识图谱特征学习模块负责；再以平滑倒词频（smooth inverse frequency) 为权重，加权平均计算得到句子向量；最后计算两个句子之间向量的夹角，获得余弦距离，得到文本相似度，由缺陷报告重复性检测模块实现。

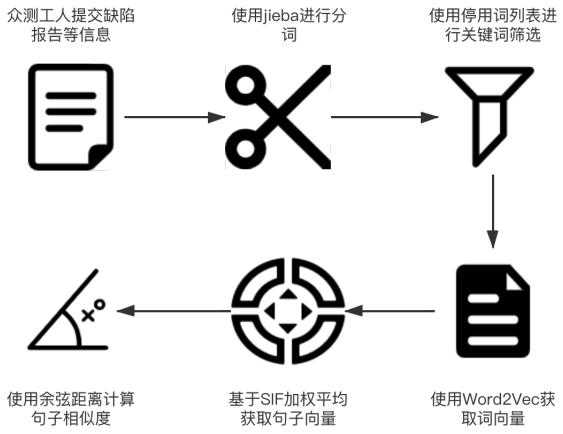

图 3-13: 文本相似度算法流程图

基于 SIF 加权平均的句向量生成方法是一种完全无监督的句向量合成方法，计算句子中词向量的加权平均作为句子的向量表示。首先进行词频的统计，频率越低的词在当前句子出现，说明它在句子中的重要性更大，也就是加权系数更大；然后将这些词向量分别减去他们各自在句向量矩阵（由词向量组合而成）的第一主向量上的投影，可以理解为删除词向量的共有部分，保留每个词向量各自拥有的特征；最后根据词频进行加权平均计算，得到句子向量。

基于 SIF 加权平均的句向量生成方法在各种文本相似度任务上的性能显著优于未加权平均；使用来自不同语料库的词频进行权重分析不会损害性能，拥有很强的鲁棒性；同时调节 a 的参数范围可以获得接近最佳的结果，获得比未加权平均值更大的改进，因此选用该方法进行句子向量的获得。

每个词向量的权重大小计算如公式3-5所示，其中 a 是平滑参数，是一个

常数，$p(w)$ 代表词频。

$$\text{Weight}_{w} = \frac{a}{a + p(w)} \tag{3-5}$$

余弦值计算公式如公式3-6所示，其中 A，B 分别代表两个向量。

$$\cos\theta = \frac{\sum_{i=1}^{n}(A_i \times B_i)}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} = \frac{A \cdot B}{|A| \times |B|} \tag{3-6}$$

获取句向量表示后，使用余弦距离度量两个句子的相似度，根据向量坐标，在高维空间中求得两个向量夹角的余弦值，余弦值越接近 1，则说明向量夹角越接近 0 度，即两向量越相似。

3.4 知识图谱构建模块详细设计

知识图谱构建模块的功能性需求主要包括众测工人填写测试报告，测试用例，缺陷报告相关信息，对缺陷报告进行查看，审核，克隆和优化，系统实时存储用户提交数据和用户行为数据，进行知识图谱的构建，用于缺陷报告的推荐和重复性检测。

3.4.1 流程设计

知识图谱构建模块流程图如图3-14所示，主要分为用户提交数据和系统进行反馈两部分，通过用户与系统之间的交互实现良性循环。

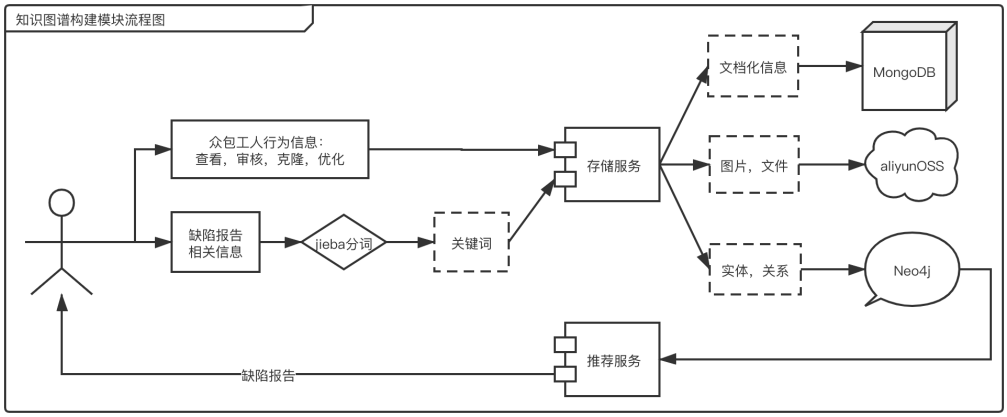

图 3-14: 知识图谱构建模块流程图

众包工人的行为信息，测试报告，测试用例，缺陷报告的相关信息和经过 jieba 分词得到的关键词均会被系统进行收集和存储。存储模块将数据分为三类，文档类信息，如缺陷报告，测试报告，测试用例等描述信息被存储至 MongoDB，将数据作为文档存储；图片，文件等大数据量信息被存储至阿里云 OSS，仅将路径存储至数据库；系统会进行实体识别，得到测试目标 Case，测试报告 Report，测试用例 TestCase，缺陷报告 Bug，众测工人 Worker，关键词 KeyWord 等实体，其中 KeyWord 实体是根据描述文字分词，然后进行同义词替换得到，将多个同义词与一个实体进行对应，最后根据真实世界逻辑关系定义实体之间的关系，将实体以及它们之间的关系存储至 Neo4j 图数据库，进行知识图谱的构建。

同时，推荐模块在众包测试过程中对众包工人进行缺陷报告推荐，众包工人可以进行查看，审核，克隆和优化，利用群体智能减少重复报告，提高报告质量。这些行为信息又会被存储模块存储，成为优化知识图谱特征学习模块和推荐模块的数据来源，实现良性循环。用户参与的众测任务越多，提交的缺陷报告越多，与系统的交互越多，推荐结果就会更加精准。

3.4.2 核心类设计

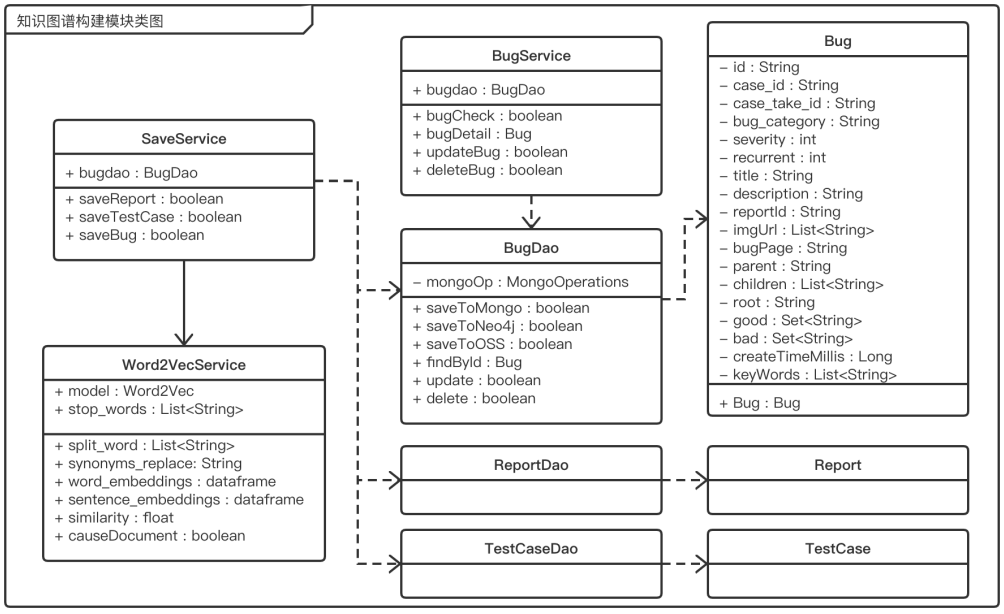

图 3–15: 知识图谱构建模块类图

知识图谱构建模块核心类图如图3-15所示。知识图谱构建模块的核心是用户提交报告，系统构建知识图谱进行反馈，用户协作发挥群体智能力量的良性循环，因此主要分为构建知识图谱和群体智能两部分。

知识图谱构建部分，SaveService 类对外提供三个接口，包括保存测试报告（saveReport），保存测试用例（saveTestCase），保存缺陷报告（saveBug）。知识图谱构建的大部分数据来自缺陷报告，缺陷报告的保存由 BugDao 具体实现，包括保存文档数据至 Mongo 数据库（saveToMongo），保存实体和关系数据至 Neo4j 数据库（saveToNeo4j），保存图片和文件数据至阿里云 OSS（uploadToOSS）。在将数据保存到 Neo4j 数据库之前，需要进行实体识别，还要对缺陷报告进行分词，同义词替换，得到关键词实体，SaveService 依赖 Word2VectorService 提供的 split_word 方法，使用 jieba 进行分词和关键词筛选，使用 synonyms_replace 方法进行同义词替换，将多个同义关键词映射为一个实体。ReportDao 和 TestCaseDao 分别负责测试报告和测试用例的保存。

群体智能部分，BugService 对外提供审核缺陷报告（bugCheck）和获取缺陷报告具体信息（bugDetail）的接口，BugDao 提供缺陷报告查找（findById），更新 (update) 和删除（delete）方法来支持 BugService。Bug，Report，TestCase 均为实体类，分别表示缺陷报告，测试报告和测试用例。

3.5 知识图谱特征学习模块详细设计

知识图谱构建完成后，系统需要进行知识图谱特征模型的构建与训练，获取实体和关系的特征，作为推荐和重复率检测模型的输入来源。

3.5.1 流程设计

知识图谱特征学习模块流程图如图3-16所示，主要包括知识图谱特征学习模型 TransD 和词向量模型 Word2Vec 的训练和使用。

首先是使用历史数据进行模型的训练，使用历史缺陷报告的关键词数据进行 Word2Vec 模型的训练，使用历史知识图谱数据进行 TransD 模型的训练。训练完成后在众包测试过程中进行模型的加载和使用。

众包测试过程中，系统收集数据进行知识图谱的构建，其中缺陷报告的关键词，会经过 Word2Vec 模型转化为词向量，作为重复性检测算法和推荐算法的输入。知识图谱中的各个实体，它们之间的关系，以及与每个实体距离在一

跳之内的邻居实体所形成的子图会作为输入，经过 TransD 模型得到实体向量，关系向量和上下文实体向量。这些特征向量会作为推荐模型的输入。

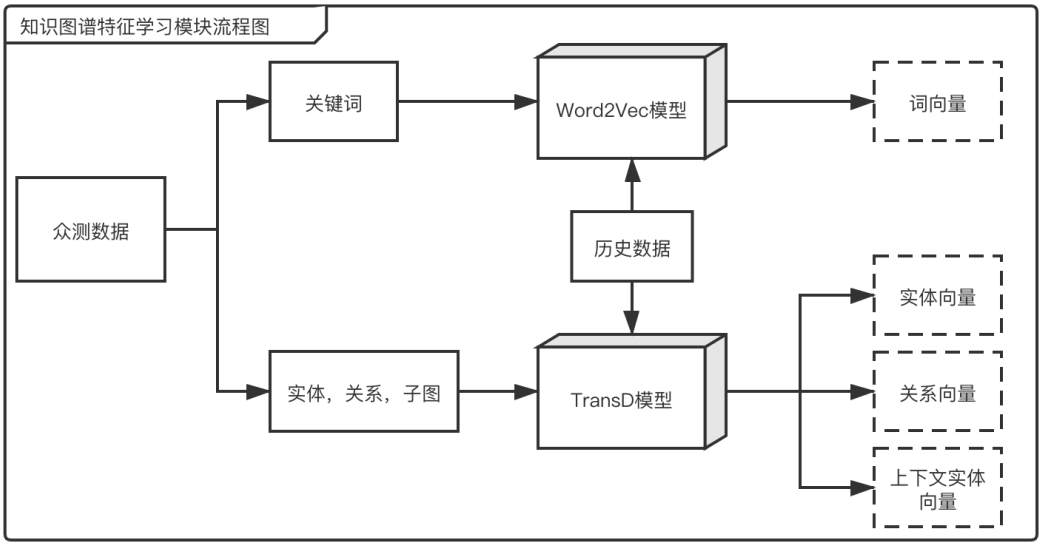

图 3-16: 知识图谱特征学习模块流程图

3.5.2 核心类设计

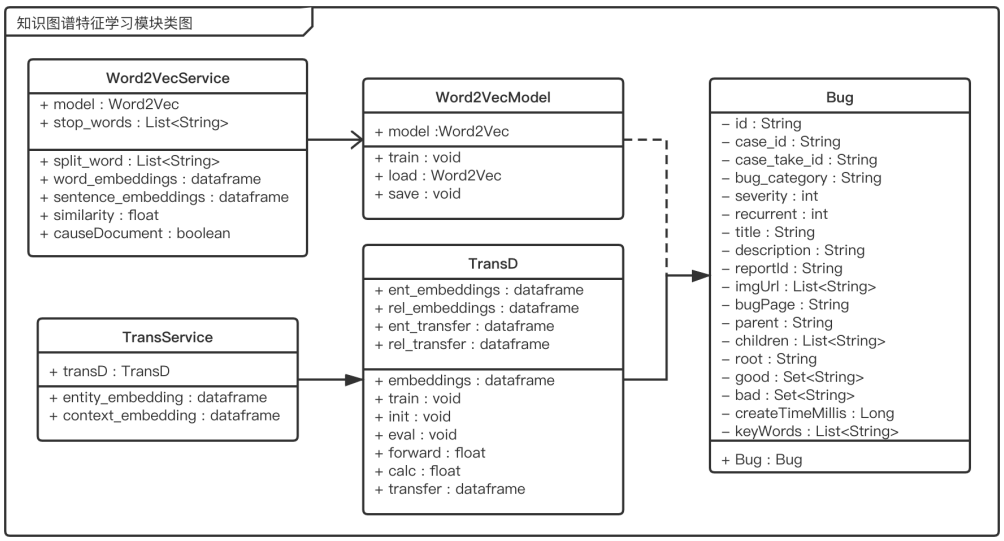

图 3-17: 知识图谱特征学习模块类图

知识图谱特征学习模块的核心类图如图3-17所示。知识图谱特征学习模块通过历史关键词数据训练 Word2Vec 模型，获取关键词对应词向量，通过历史实体和关系数据训练 TransD 模型，获取词语对应的实体向量，关系向量和上下文特征向量。

实体向量获取方面，TransService 对外提供实体向量接口（entity_embeddings）和实体上下文向量接口（context_embeddings），依赖于 TransD 类进行模型的训练和向量的转换。TransD 类包含属性实体向量 ent_embeddings，关系向量 rel_embeddings，实体转换矩阵 ent_transfer 和关系转换矩阵 rel_transfer，提供获取实体向量（embeddings），属性初始化（init），训练模型（train），前向传播（forward），实体和关系投影（transfer），计算误差（calc）等方法。

词向量获取方面，Word2VectorService 对外提供词向量接口（word_embeddings），依赖于 Word2VecModel 类进行模型的训练和向量的转换。Word2VecModel 类提供训练模型（train），加载模型（load），保存模型（save）等方法。

3.6 缺陷报告推荐模块详细设计

缺陷报告推荐模块的功能性需求包括缺陷报告推荐，在用户编辑缺陷报告，进行各种选择和内容填写的过程中，系统会根据已有的报告数据和已构建的知识图谱，进行缺陷报告的推荐，供用户进行查看，审核和优化。随着报告属性的不断补全和文字的输入，系统推荐的报告会越来越精准，这使得用户可以了解他人的进度和成果，不去挖掘已知的 Bug，减少重复报告的产生，同时进行群体协作，对已有的缺陷报告进行补充描述，提高报告质量和众测效率。

3.6.1 流程设计

缺陷报告推荐模块流程图如图3-18所示。主要使用 DKN 模型进行缺陷报告的推荐。DKN 模型采用依次学习将知识图谱作为推荐系统的辅助信息加以利用，知识图谱特征学习模块和推荐系统模块相互独立。其中知识图谱特征学习模块已在 4.2 节进行了详细论述，此处不再赘述。

输入包括历史缺陷报告数据，候选缺陷报告数据和标签 label，表示用户是否关注该历史报告。每份缺陷报告都经过知识图谱特征学习模块的 Word2Vec 模型和 TransD 模型获取了缺陷报告的词向量，实体向量和上下文实体向量，拼接三部分向量作为 KCNN 的输入，经过维度变换，激活函数，卷积和池化，

得到缺陷报告的特征向量。将历史缺陷报告向量和候选报告向量作为 Attention Net 的输入，计算候选报告对于用户每份关注报告的 attention，再做加权求和，使用用户关注的缺陷报告向量得到用户向量表示。最后得到用户点击该份候选报告的概率。将所有候选报告的概率从大到小排序，将点击概率最高的几份报告返回给用户，作为推荐。

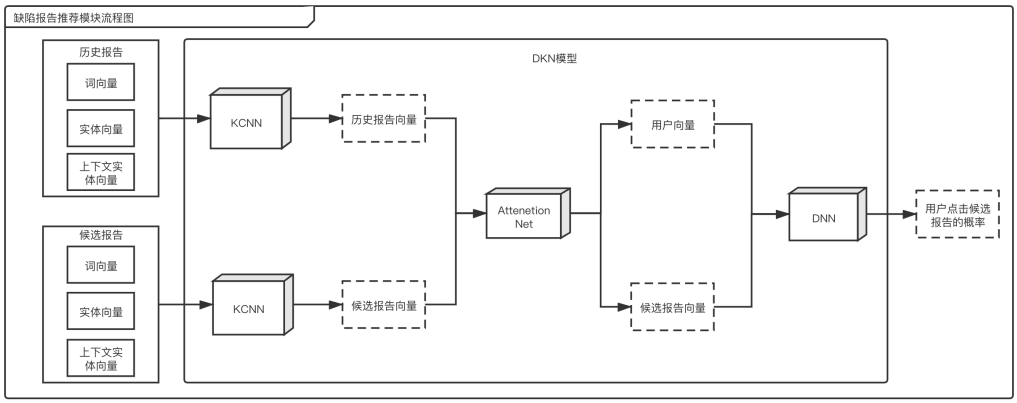

图 3-18: 缺陷报告推荐模块流程图

3.6.2 核心类设计

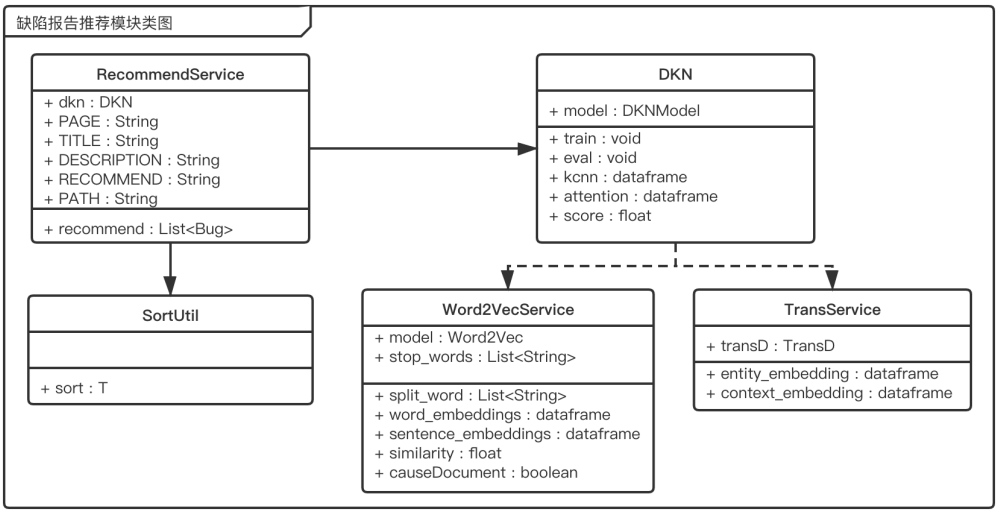

图 3-19: 缺陷报告推荐模块类图

缺陷报告推荐模块核心类图如图 3-19所示。主要是依赖 DKN 类进行缺陷报告推荐模型的训练与服务。

RecommendService 对外提供缺陷报告推荐接口（recommend），定义多个 final 属性用于查找 Redis 缓存数据，包括所处页面（PAGE），报告标题（TITLE），报告描述（DESCRIPTION），候选报告（RECOMMEND），点击路径（PATH），依赖于 DKN 类进行缺陷报告推荐。

DKN 类实际进行模型的训练和点击概率的计算，获得关键词向量，实体向量，实体上下文向量，使用 KCNN 获取缺陷报告最终向量，基于注意力机制得到用户点击缺陷报告的概率。提供训练（train），KCNN 网络定义（kcnn），注意力网络定义（attention），点击概率计算（score）等方法。DKN 类依赖 Word2VecService 类获取缺陷报告关键词向量，依赖 TransService 类获取关键词对应实体向量和上下文实体向量。

SortUtil 是工具类，对外提供排序（sort）方法，RecommendService 使用它对于缺陷报告的点击概率进行排序。

3.7 缺陷报告重复性检测模块详细设计

缺陷报告重复性检测模块的功能性需求主要是缺陷报告重复性检测。在众测任务完成后，需要组织评审专家针对众测工人提交的缺陷报告进行评分，用于报告整编。虽然在众测任务过程中，系统通过推荐和提示尽量减少众测工人提交的重复报告，但重复报告的出现仍是不可避免的，因此系统希望通过缺陷报告重复性检测模块对缺陷报告进行自动查重，评分，减少评审专家的工作量，提高评审效率，也提高最终交付报告的质量。

3.7.1 流程设计

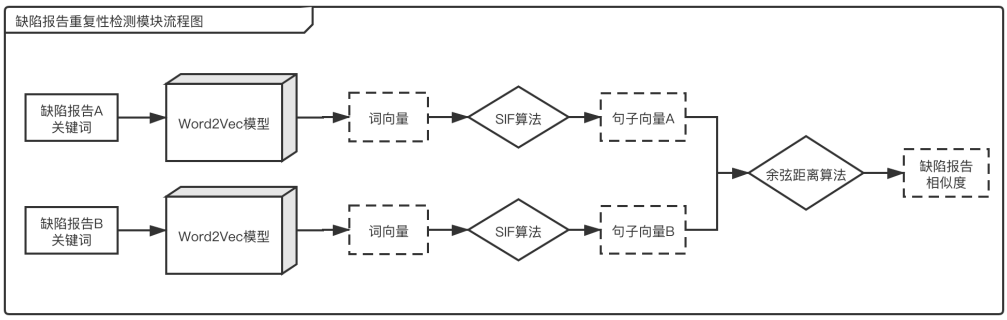

图 3-20: 缺陷报告重复性检测模块流程图

缺陷报告重复性检测模块流程图如图3-20所示。主要针对一场众测任务的缺陷报告进行重复性检测，自动置零和说明文档生成。

以两份缺陷报告为例，首先从知识图谱中获取他们的关键词，经过知识图谱特征学习模块训练好的 Word2Vec 模型获取关键词对应的词向量。根据词向量获取对应句子的句子向量，以平滑倒词频为权重，对所有词的词向量进行加权平均，然后分别减去词向量在句向量矩阵的第一主向量上的投影，保留每个词向量各自拥有的特征，获得句子向量。针对得到的句子向量，进行余弦距离的计算，最终获得两份缺陷报告之间的相似度。

与较早提交的缺陷报告相似且游离于缺陷报告树的报告，会被认为是重复报告，众测管理员会对这些报告进行人工复核，避免自动检测出现错误，核实无误后将重复报告的参考分置为零分，减少评审专家的工作量。系统也会产生说明文档，用于复盘和对众测工人进行说明。

3.7.2 核心类设计

缺陷报告重复性检测模块核心类图如图3-21所示。主要进行缺陷报告的重复性检测，自动置零分和说明文档生成。

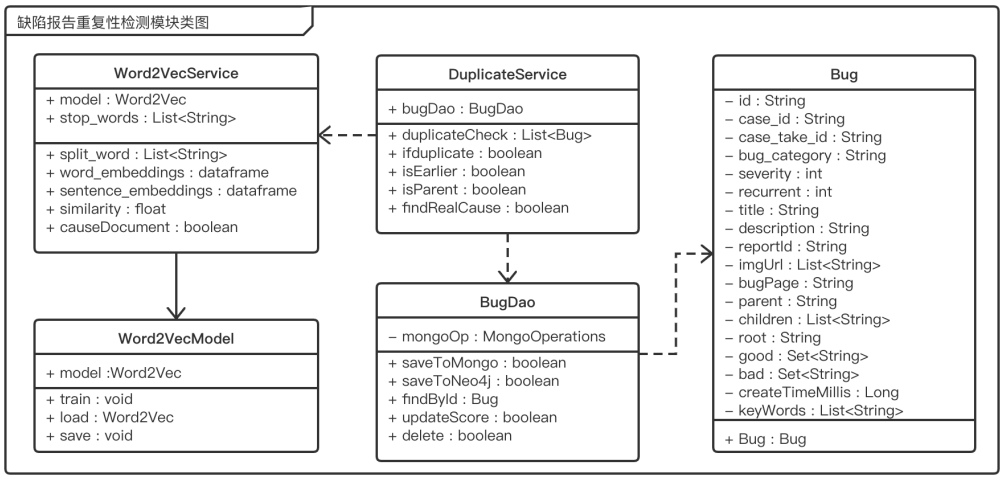

图 3-21: 缺陷报告重复性检测模块类图

DuplicateService 类对外提供众测任务整体查重（duplicateCheck）和两份缺陷报告单独查重（ifDuplicate）接口，同时还有判断提交时间是否早于（isEarlier），判断两份缺陷报告是否具有亲缘关系（isParent），查找相似度最高的

缺陷报告（findRealCause）等方法。DuplicateService 依赖 Word2VecService 类进行重复率的计算和说明文档的生成。

Word2VecService 类根据 Word2VecModel 类获取关键词向量，根据 SIF 方法得到句子向量，使用余弦距离计算两个句子之间的相似度，得到缺陷报告之间的重复性，生成重复报告说明文档。提供获取句子向量（sentence_embeddings），重复率计算（similarity）和说明文档生成（causeDocument）方法。

BugDao 提供缺陷报告的查找（fidnById）方法和分数更新方法（update_score），用于将参考分置为零分。Bug 是实体类，代表缺陷报告。

3.8 本章小结

本章针对基于知识图谱的众测智能推荐技术进行了需求分析和系统设计。首先使用用例图和用例表对系统需求进行了具体描述。接着，使用“4+1”视图从不同角度对系统的架构和实现进行了整体概要设计。然后使用类图，ER 图等对系统实体类和数据库集合进行了设计。接着，使用流程图和公式对本系统使用的推荐算法和重复率检测算法进行了说明和设计。最后使用流程图和核心类图对系统的四个模块：知识图谱构建模块，知识图谱特征学习模块，缺陷报告推荐模块，缺陷报告重复性检测模块进行了详细设计和说明，为第四章提供了坚实的技术基础。

第四章　基于知识图谱的众测智能推荐系统的实现

4.1　知识图谱构建模块的实现

知识图谱构建模块的功能性需求主要包括众测工人填写测试报告，测试用例，缺陷报告，系统实时存储数据并构建知识图谱，用于缺陷报告的推荐和重复性检测。该模块主要分为知识图谱构建和群体智能两部分，众测工人在执行众测任务时的所有数据都会被存储并进行知识图谱构建，系统提供的反馈又会促进众测工人群体智能的发挥，实现良性循环。

4.1.1　知识图谱构建实现

图 4-1: 缺陷报告填写截图

用户在测试平台主站主站点击“编辑报告”按钮后，系统跳转至众测前端页面，开始测试报告的填写，填写内容包括测试报告名称，测试设备品牌，测试设备名称，测试设备操作系统。然后进行测试用例的创建，填写测试用例名称，前置条件，测试步骤，预期结果。

上述步骤完成后，用户进行缺陷报告的创建，如图4-1所示。用户首先选择该 Bug 所属的三级页面，其中只有第一级页面属于必选项。接着，用户进行漏洞分类，严重程度和复现程度等 Bug 基本属性的选择，并将该缺陷报告与之前创建的测试用例进行关联。然后，用户进行缺陷报告标题和详细描述的填写，最后上传 Bug 截图，进行补充说明。用户所有提交的数据和用户本身的信息及行为信息，都会成为知识图谱构建的数据来源。

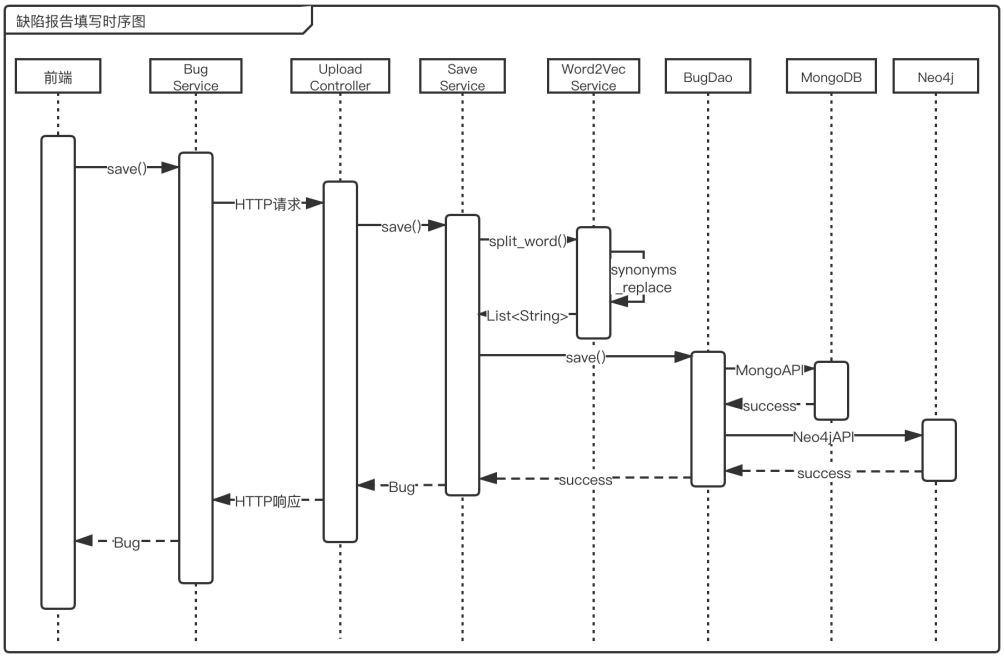

图 4-2: 缺陷报告填写时序图

众测工人填写缺陷报告的时序图如图4-2所示，用户点击报告保存按钮后，触发前端 confirmSaveBug 方法，调用 BugService 的 save 方法，向后端发送 POST 请求，传输缺陷报告详细信息，UploadController 的 submit 方法获取缺陷报告详细信息，调用 SaveService 的 save 方法，该方法调用 split_word 方法提取缺陷报告的关键词，调用 synonyms_replace 方法进行同义词替换，将多个同义关键词映射为一个实体，然后调用 BugDao 的 save 方法进行数据存储，该方法首先将文档数据存储至 MongoDB 数据库，包括缺陷报告基本信息，缺陷报

告与测试用例之间的对应关系，缺陷报告之间的父子关系等，然后将文件数据如截图等存储至 OSS，最后将知识图谱数据存储至 Neo4j 数据库，包括实体，实体之间的关系，新增实体与已有实体之间的关系，进行知识图谱的构建。

以一场 2020 年 4 月真实的众测任务“咕咚翻译 APP”为例，构建的知识图谱可视化后如图 4-3所示，该场众测任务共 88 人参加，提交了 88 份测试报告，423 个测试用例，401 份缺陷报告，受限于展示界面大小，本文只展示了知识图谱的局部。可以看到，最中间的棕色节点是测试目标 Case，他与多个红色 Report 节点之间存在 inculde 关系，每一个 Report 节点都与 0 到 N 个橙色 TestCase 节点之间存在 include 关系，而每一个 TestCase 节点均与 0 到 N 个蓝色 Bug 节点之间存在 include 关系。

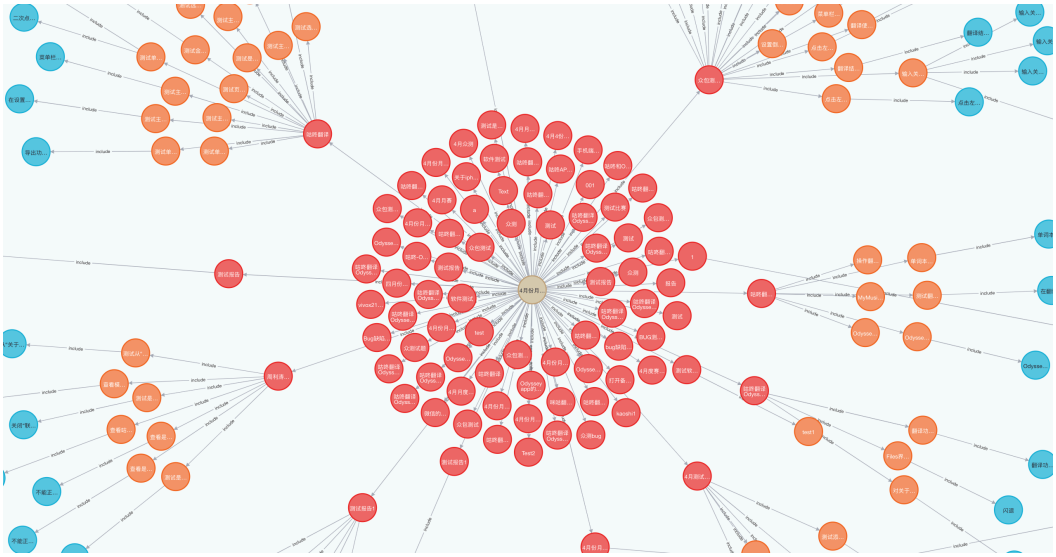

图 4-3: 知识图谱可视化

在存储缺陷报告数据的过程中，需要进行信息抽取，获取关键实体和他们之间的关系。系统需要存储缺陷报告文字描述中所含关键词对应的实体，用于知识图谱的构建和模型的训练，因此需要对文字进行分词，获取关键词，然后对同义词进行替换，将多个同义关键词映射为同一个实体。

分词代码如图 4-4所示。首先读取 ChineStopWords.txt 文件，创建停用词列表，该文件共 3496 行，存储了包括标点符号，特殊字符，助词、副词、介词、连词等多种无用词在内的停用词，比如逗号，句号，“的地得”，“我”等等，去除这些词语，可以节省存储空间，提高搜索推荐效率，减少无用词语对于文本关键词的干扰。接着，使用开源工具 jieba 分词的精确模式进行分词。最后，对于分词得到的词语列表进行停用词的去除，得到缺陷报告的关键词。

```
def split_words(description):
    # 读取文件，创建停用词列表
    stopwords = [line.strip() for line in open('ChineseStopWords.txt', encoding='UTF-8').readlines()]
    # 使用 jieba 进行分词
    split_word = [x for x in jieba.cut(description, cut_all=False) if x != '']
    # 去除停用词
    for word in split_word:
        if word in stopwords:
            split_word.remove(word)
    return split_word
```

图 4-4: 文字描述分词代码

4.1.2 群体智能实现

为了实现群体智能，众包工人在填写的过程中会不断得到缺陷报告的推荐，也可以通过主动搜索查找感兴趣的缺陷报告。缺陷报告列表中只展现了缺陷报告的粗略信息，包括标题，漏洞分类，截图，推荐度，点击列表中任意一条缺陷报告可以看到缺陷报告的详细信息，如图 4-5所示，缺陷报告具体信息包括报告 ID，所处位置，基本属性，创建时间，标题，描述，截图等。

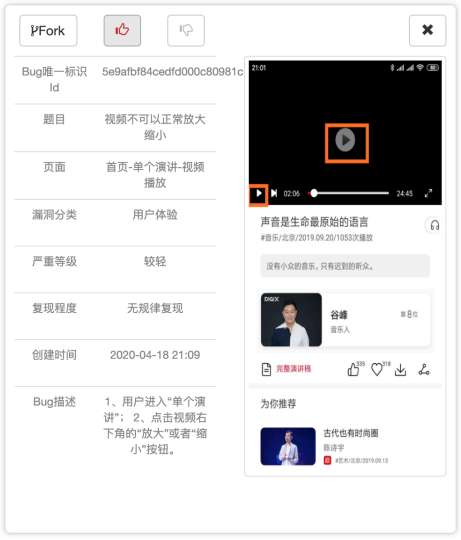

图 4-5: 查看缺陷报告具体信息截图

众包工人可以点击“点赞”按钮进行点赞，点击“点踩”按钮进行点踩，表达对这份缺陷报告的赞同或不赞同，用户的这些审核记录会对报告整编时缺陷报

告的优先级和可信度有一定影响，决定了最终交付报告的质量。

如果众包工人觉得一份缺陷报告写得不够完整，可以点击“Fork”按钮进行报告克隆，将该缺陷报告进行复制，如图4-6所示，新报告与被克隆报告完全一致，除了三级页面不可修改外其他信息都可以进行修改，众包工人在此基础上进行修改和优化，提交后新报告作为旧报告的子报告进行保存，提高针对某个 Bug 的描述质量。

同时，众包工人无论是查找缺陷报告，点击查看缺陷报告具体信息，审核缺陷报告，还是克隆缺陷报告，均被系统认为众包工人对该缺陷报告进行了关注，系统会对众包工人的行为信息进行记录，用于缺陷报告推荐。

图 4-6: 克隆优化缺陷报告截图

众测工人群体智能的时序图如图4-7所示，用户点击缺陷报告列表中的一份报告，触发前端 getBugDetail 方法，调用 BugService 的 getDetail 方法，向后端发送 GET 请求，BugController 获取 BugId，调用 BugService 的 getDetail 方法，该方法调用 BugDao 的 findById 方法，以 ID 为条件查找 Bug，将信息返回给前端展示。用户进行点赞或点踩，触发前端 check 方法，向后端发送 POST 请求，BugController 获取 BugId 和用户操作，调用 BugService 的 bugCheck 方法，该方法更新 Bug 信息，添加点赞点踩记录，调用 BugDao 的 save 方法进行数据更新。克隆缺陷报告将被克隆报告的信息填充至缺陷报告编辑框，是完全的前端操作，用户点击克隆后，系统将属性赋值给填写报告的变量，在前端页面进行展示，用户在此基础上进行编辑，与新建一份缺陷报告的流程一致。

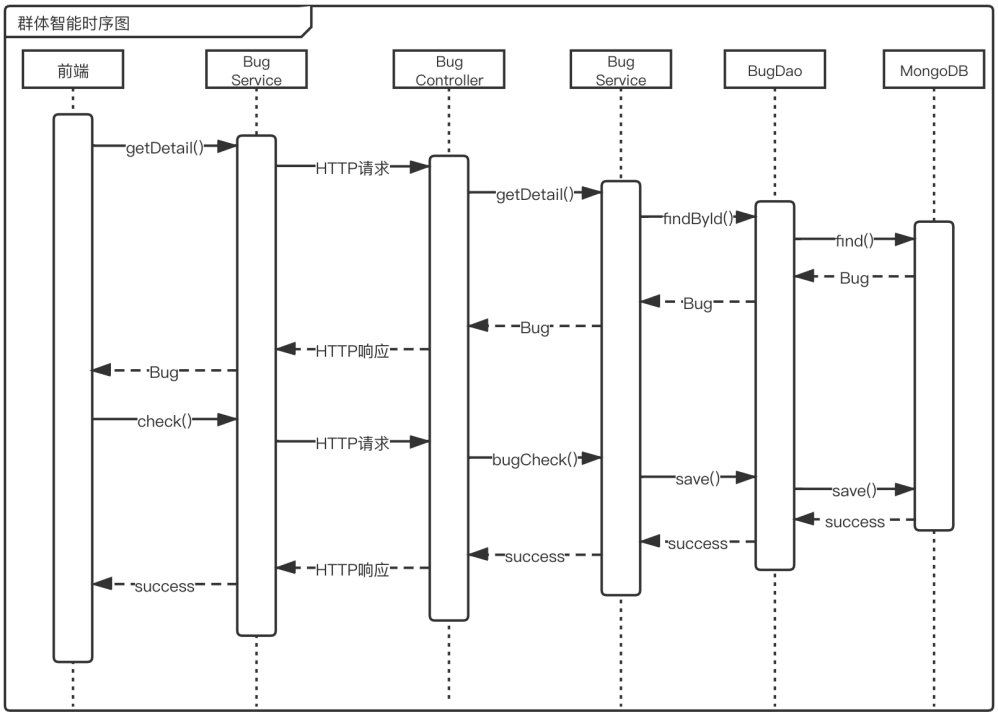

图 4-7: 群体智能时序图

```
public void saveGood(String bugId, String reportId) {
    //是否评价过
    if (bugDao.haveJudged(bugId, reportId)) {
        //保存给缺陷报告点赞的测试报告 ID
        bugDao.saveGood(bugId, reportId);
        //保存测试报告点赞的缺陷报告 ID
        reportDao.saveGood(bugId, reportId);
    }
}
public boolean haveJudged(String bugId, String reportId) {
    //省略查找语句
    //缺陷报告被点赞点踩列表中是否存在该测试报告 ID
    if (bug.getGood().contains(reportId) || bug.getBad().contains(reportId)) {
        return true;
    }
    return false;
}
```

图 4-8: 审核缺陷报告代码

缺陷报告审核代码如图4-8所示，以点赞为例，输入参数是缺陷报告 ID 和

测试报告 ID。首先调用 BugDao 的 haveJudged 方法，确定该缺陷报告未被该用户审核过，具体实现是使用 MongoOperations，通过 bugId 查找对应缺陷报告，查看缺陷报告的被审核列表中是否存在该测试报告的 ID，如果存在说明该用户已经审核过该缺陷报告，目前操作是不合理的，返回 False，反之未被审核过，调用 BugDao 的 saveGood 方法保存给缺陷报告点赞的测试报告 ID，调用 ReportDao 的 saveGood 方法保存该测试报告点赞的缺陷报告 ID，缺陷报告与测试报告之间是一个多对多关系，系统进行冗余保存便于检索查找。保存测试报告点赞的缺陷报告 ID 列表是为了与其他信息进行汇总，得到用户关注过的缺陷报告，用于缺陷报告推荐。

4.2 知识图谱特征学习模块的实现

知识图谱构建完成后，系统需要进行知识图谱特征模型的构建与训练，获取实体和关系的特征，作为推荐和重复率检测模型的输入数据。

4.2.1 重复率检测模型特征学习实现

本文使用 Word2Vec 对于文本词语的特征进行学习，将文本转换为词向量。Word2Vec 模型训练代码如图 4-9所示，使用开源 Gensim 库进行训练。

```
# word2vec 模型训练
def word2vec_trans_model(bug_data):
    # 省略从缺陷报告获取分词结果的代码
    # 词向量为 200 维，采用 5 个线程训练，采用 skip-gram 模型
    model = Word2Vec(size=200, workers=5, sg=1)
    model.build_vocab(cut_descriptions)
    # 训练模型
    model.train(cut_descriptions,total_examples=model.corpus_count,
                epochs=model.iter)
    # 保存模型
    model.save('model/bug_data_model')
```

图 4-9: word2vec 模型训练代码

首先从缺陷报告数据中获取文字描述分词后的结果，分词代码可见 4-4。考虑到语料来自于经过筛选的历史缺陷报告，充分且较为准确，因此选用 Skip-Gram 模型，给定中心词，对上下文进行预测。默认考虑中心词前后各五

个单词，因为众测平台历史数据较多，语料较大，设定词向量为 200 维，设定训练线程为 5，进行模型的构建。调用 build_vocab 方法从句子中建立词汇表，然后调用 train 方法进行模型的训练。训练完成后，调用 save 方法将训练好的模型进行保存，保存后的模型可以直接使用 load 方法进行加载和使用。在众测过程中将缺陷报告的关键词作为模型的输入，获得词语对应的词向量。

4.2.2 知识图谱特征学习实现

如 3.3.1 节对于推荐算法的介绍，DKN 模型中推荐系统的输入包括缺陷报告文本的词向量，词语对应实体的向量和实体上下文向量。词向量通过 Word2Vec 模型获得，实体向量和实体上下文向量通过 TransD 模型获得。

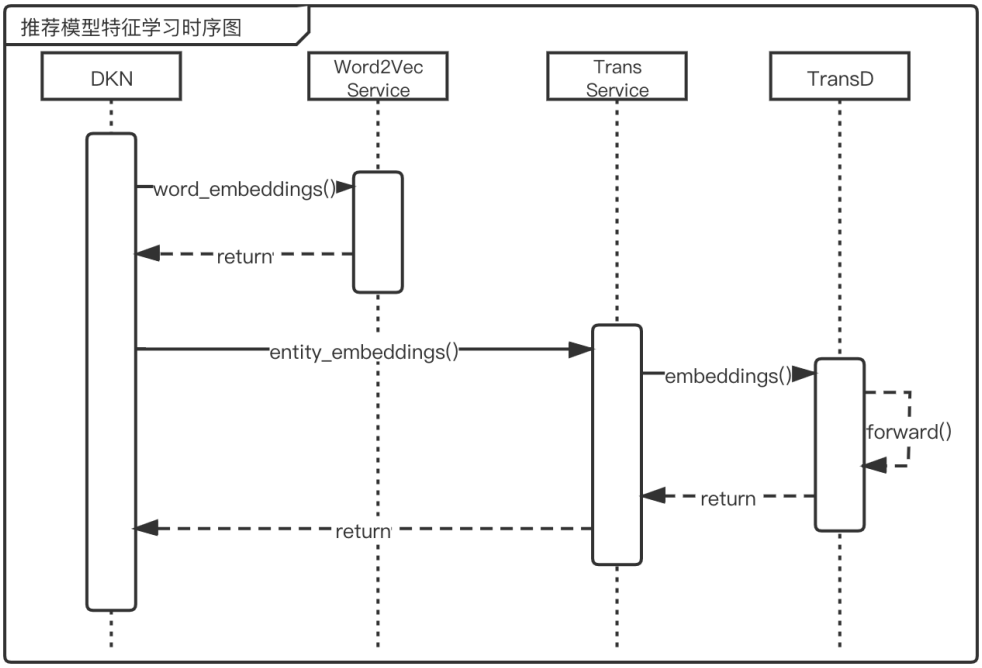

图 4-10: 知识图谱特征学习时序图

知识图谱特征学习的顺序图如图4-10所示。DKN 类的 build_input 方法通过调用 Word2VecService 的 word_embeddings 方法获取文本的词向量，该方法需要使用 Word2Vec 模型，关于该模型的训练已于 4.2.1 节进行介绍。通过调用 TransService 的 entity_embeddings 获得实体向量，该方法调用 TransD 类的 embeddings() 方法，使用 TransD 这种基于距离的翻译模型，将头尾实体投影到

关系空间，获取实体和关系向量。实体上下文向量可以通过计算邻居实体向量的算术平均值获得。

TransD 模型的输入主要是实体索引文件，关系索引文件，三元组表示文件。模型训练中最核心的代码是神经网络中代表前向传播的方法 forward，如图4-11所示。

```
# 前向传播函数 forward，获取头尾实体以及关系的 Embedding
def forward(self, data):
    # 省略__init__方法
    head = data['head']
    tail = data['tail']
    relation = data['relation']
    mode = data['mode']
    head_embeddings = self.ent_embeddings(head)
    tail_embeddings = self.ent_embeddings(tail)
    relation_embeddings = self.rel_embeddings(relation)
    # 实体和关系不同的空间表示
    head_transfer = self.ent_transfer(head)
    tail_transfer = self.ent_transfer(tail)
    rel_transfer = self.rel_transfer(relation)
    # 计算投影矩阵和头尾实体投影向量
    head_embeddings = self.transfer(head_embeddings, head_transfer, rel_transfer)
    tail_embeddings = self.transfer(tail_embeddings, tail_transfer, rel_transfer)
    # 计算 h+r-t，期望为 0
    score=self.calc(head_embeddings, tail_embeddings, relation_embeddings, mode)
    return score
```

图 4-11: 前向传播函数 forward

首先在 init 方法中定义了四个 Embedding 矩阵并初始化。ent_embeddings 和 rel_embeddings 分别代表实体 Embedding 和关系 Embedding，ent_transfer 和 rel_transfer 分别代表实体转换矩阵和关系转换矩阵。如公式3-1和3-2所示，首先计算头尾实体投影矩阵，然后调用 transfer 方法，使用公式3-3将头尾实体投影至关系空间，计算头尾实体的投影向量，最后调用 calc 方法，计算损失 score=(h+r-t) 的大小，即头实体向量加上关系向量与尾实体向量的差值，在不断的循环训练中使得 score 越来越小，获得最优的神经网络参数，得到可用的模型。在众测过程中加载训练好的模型，将缺陷报告中存在的实体作为模型的输入，获得实体的向量和上下文向量。

4.3 缺陷报告推荐模块的实现

在用户编辑缺陷报告，进行各种选择和内容填写的过程中，系统会根据用户的历史数据，已有的报告数据和已构建的知识图谱，进行缺陷报告的推荐，供用户进行查看，审核，克隆和优化。随着报告属性的不断补全和用户与系统更多的交互，系统推荐的报告会越来越精准，这使得用户可以了解他人的进度和成果，减少重复报告的产生，同时进行群体协作，提高报告质量和众测效率。推荐报告列表如图4-12所示，展示了缺陷报告的标题，漏洞分类，截图和相似度，便于用户粗略了解缺陷报告相关信息。

图 4-12: 缺陷报告推荐列表截图

4.3.1 缺陷报告推荐顺序设计

推荐缺陷报告的时序图如图4-13所示。每当用户修改三级页面和 Bug 基本属性的选择，或是修改缺陷报告的标题和文本描述，都会触发前端的 getRecommendList 方法，向后端服务发送 POST 请求，传输修改类型和修改内容，后端 RecommendController 获取数据后，调用 RecommendService 的 recommend 方法进行报告推荐。该方法更新 Redis 缓存中保存的用户当前正在编辑的缺陷报告数据，并基于知识图谱特征学习模块训练的模型获取缺陷报告对应的向量表示，包括关键词向量，实体向量和实体上下文向量，通过三

级页面筛选获得候选推荐报告集合，也获取上述三种向量。调用 DKN 类的 recommend 方法获得推荐列表，该方法调用 kcnn 方法获得用户关注报告和候选缺陷报告的向量表示，再调用 attention 方法，采用基于注意力机制的用户兴趣预测方法得到用户向量，最终调用 score 方法计算用户点击某份缺陷报告的概率值。RecommendService 最后调用 SortUtil 的 sort 方法按概率值从大到小进行排序，向前端返回前 6 份最可能被点击的缺陷报告，进行展示。

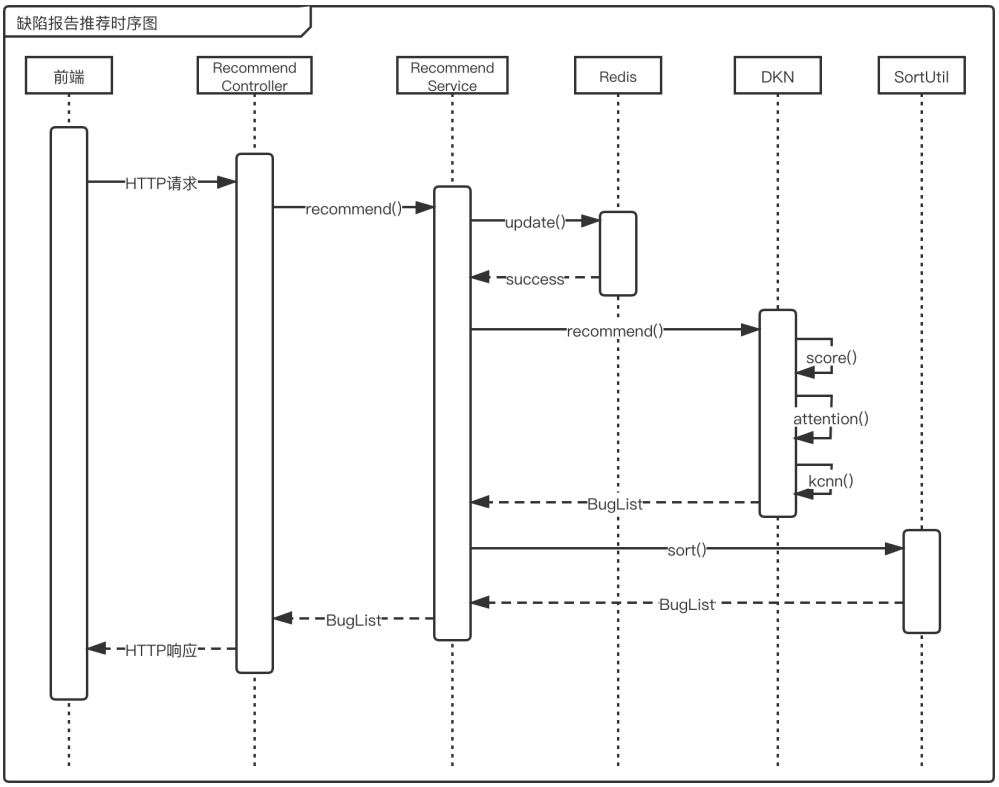

图 4-13: 缺陷报告推荐时序图

4.3.2 缺陷报告推荐关键代码

在知识图谱特征学习模块，本文进行了 TransD 模型和 Word2Vec 模型的构建与训练，这属于知识图谱推荐过程中的知识提取部分，根据 Word2Vec 模型，可以获得缺陷报告中每个词语的向量表示；根据 TransD 模型，可以得到每个词语对应的实体的向量表示，再根据公式 3-4可以得到每个词语的上下文特征。DKN 模型的输入包括历史报告关键词、历史报告实体、候选集报告的关键词、候选集报告的实体、用户是否关注过该历史报告（label），模型的训练代

码的关键代码包括 kcnn 方法和 attention 方法，如下所示。

kcnn 方法代码如图 4-14所示。首先调用 TransD 和 Word2Vec 模型获得缺陷报告关键词，关键词对应实体，实体上下文的向量。调用 tf.concat 方法拼接三部分向量作为 KCNN 的输入，此时出现了变量维度的变换。接着进行卷积，调用 tf.conv2d 函数输入张量，设定卷积核，调用 tf.nn.relu 函数设定激活函数为 relu，该函数将小于 0 的数置为 0，大于 0 的数则保持不变。然后进行池化，调用 tf.nn.max_pool 函数进行最大值池化。最后获得缺陷报告的向量表示。用户关注过的报告和候选报告均需通过 KCNN 获取向量表示。

```
def _kcnn(self, words, entities, args):
    # 省略获取输入的代码
    # 获取关键词和实体向量
    embedded_words = tf.nn.embedding_lookup(self.word_embeddings, words)
    embedded_entities = tf.nn.embedding_lookup(self.entity_embeddings, entities)
    # 考虑上下文，获取上下文实体向量，进行三部分向量的拼接
    embedded_contexts= tf.nn.embedding_lookup(self.context_embeddings, entities)
    concat_input=tf.concat([embedded_words,embedded_entities,
                    embedded_contexts], axis=-1)
    full_dim = args.word_dim + args.entity_dim * 2
    concat_input = tf.expand_dims(concat_input, -1)
    outputs = []
    for filter_size in args.filter_sizes:
        # 省略参数初始化代码
        # 卷积
        conv=tf.nn.conv2d(concat_input, w, strides=[1, 1, 1, 1], padding='VALID',
                name='conv')
        relu = tf.nn.relu(tf.nn.bias_add(conv, b), name='relu')
        # max pooling 池化
        pool = tf.nn.max_pool(relu, ksize=[1, args.max_title_length - filter_size + 1, 1,
                1],strides=[1, 1, 1, 1], padding='VALID', name='pool')
        outputs.append(pool)
    output = tf.concat(outputs, axis=-1)
    output = tf.reshape(output, [-1, args.n_filters * len(args.filter_sizes)])
    return output
```

图 4-14: kcnn 代码

注意力机制代码 attention 方法如图 4-15所示，首先调用 kcnn 方法获得历史报告和候选报告的向量，调用 tf.reshape 方法对历史报告向量进行维度变换，调用 tf.expand_dims 方法将候选报告向量添加一个维度。然后调用

tf.reduce_sum 方法建立句子序列中的每个词与句子其他词的注意力权重，接着调用 tf.nn.softmax 方法将注意力权重向量进行 softmax 归一化，最后调用 tf.reduce_sum 方法进行加权求和得到 attention，作为用户的向量表示。最后调用 tf.sigmoid 方法计算 score，作为用户点击该候选报告的概率。

```
# 注意力机制代码
def _attention(self, args):
    # 省略获取关键词向量和实体向量
    # 调用 kcnn 获得 embeddings
    with tf.variable_scope('kcnn', reuse=tf.AUTO_REUSE):
        focused_embeddings = self._kcnn(focused_words, focused_entities, args)
        candidate_embeddings=self._kcnn(candidate_words,candidate_entities, args)
    # 维度变换
    focused_embeddings=tf.reshape(focused_embeddings,shape=[-1,
                        args.max_click_history, args.n_filters * len(args.filter_sizes)])
    # 增加维度
    news_embeddings_expanded = tf.expand_dims(candidate_embeddings, 1)
    # 建立注意力权重
    attention_weights=tf.reduce_sum(focused_embeddings*
                        news_embeddings_expanded, axis=-1)
    # softmax 归一化
    attention_weights = tf.nn.softmax(attention_weights, dim=-1)
    attention_weights_expanded = tf.expand_dims(attention_weights, axis=-1)
    # 加权求和
    user_embeddings=tf.reduce_sum(focused_embeddings*
                        attention_weights_expanded, axis=1)
    return user_embeddings, candidate_embeddings

user_embeddings, news_embeddings = self._attention(args)
self.score_unnormalized=tf.reduce_sum(user_embeddings*news_embeddings,axis=1)
self.score = tf.sigmoid(self.score_unnormalized)
```

图 4-15: attention 机制代码

4.4 缺陷报告重复性检测模块的实现

在众测任务完成后，需要组织评审专家针对众测工人提交的缺陷报告进行评分。虽然在众测任务过程中，系统通过推荐和提示尽量减少众测工人提交的重复报告，但重复报告的出现仍是不可避免的，因此通过缺陷报告重复性检测模块对缺陷报告进行自动查重，提高评审效率和最终交付报告的质量。

4.4.1 缺陷报告重复性检测顺序设计

缺陷报告重复性检测的时序图如图4-16所示。众测管理员在众测任务列表中选择需要查重的众测任务，输入重复率要求，点击查重，触发前端 BugService 的 duplicateCheck 方法，向后端服务发送 POST 请求，DuplicateController 获取众测任务 ID，调用 DuplicateService 的 duplicateCheck 方法，根据众测个任务 ID 获取该众测任务中提交的缺陷报告，对同一三级页面下的缺陷报告进行重复性检测。在众测管理员人工复核，确认无误后，调用 BugDao 的 save 方法将重复报告评分保存为 0 分，同时调用 causeDocument 方法生成置零说明文档，供用户下载查看，向前端返回结果和说明。

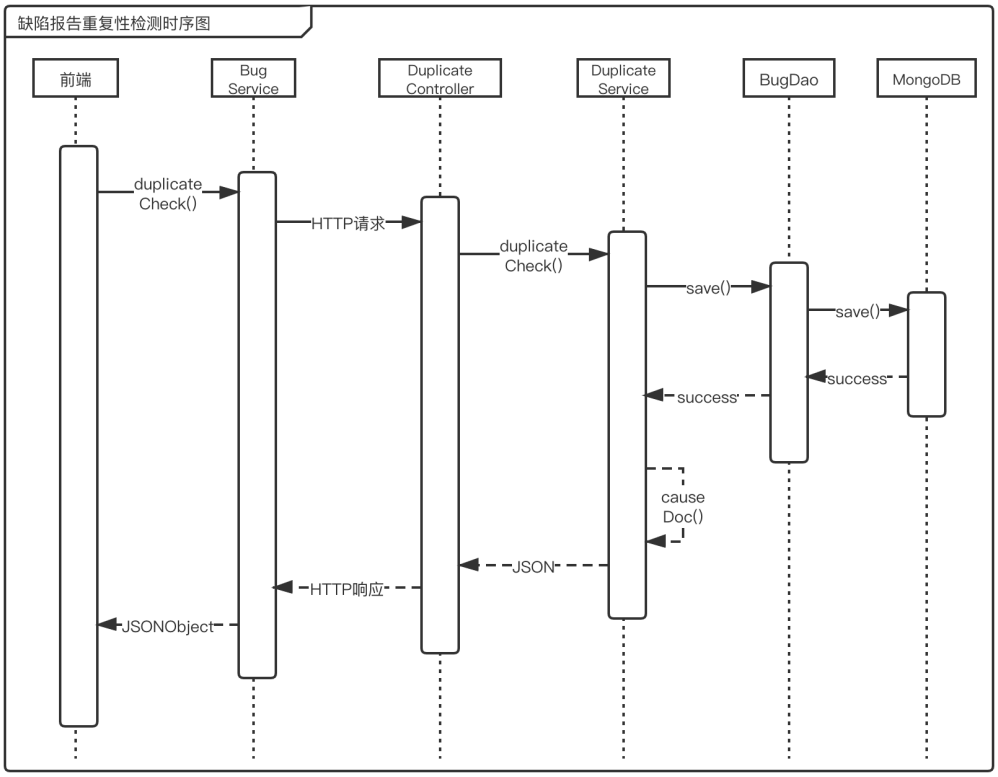

图 4-16: 缺陷报告重复性检测时序图

4.4.2 缺陷报告重复性检测关键代码

在众测任务完成后，系统会自动进行 Word2Vec 模型的更新和训练，获取缺陷报告标题和描述的词向量，具体实现已在上文进行了详细描述，此处不再赘述。计算文字相似度时首先调用预训练好的模型，调用 sentence_vector 方法

获取句子转换得到的向量表示，调用 cos_distance 方法计算两个向量之间的余弦距离，获取句子之间的相似度。

```
# 加权计算
def get_weighted_average(word_vector, index, weight):
    n_samples = index.shape[0]
    emb = np.zeros((n_samples, word_vector.shape[1]))
    for i in xrange(n_samples):
        emb[i,:]=weight[i,:].dot(word_vector[index[i,:],:])/
                np.count_nonzero(weight[i, :])
    return emb

# 计算主成分
def compute_pc(emb):
    svd = TruncatedSVD(n_components=1, n_iter=7, random_state=0)
    svd.fit(emb)
    return svd.components_

# 移除主成分
def remove_pc(emb):
    pc = compute_pc(emb)
    emb = emb - emb.dot(pc.transpose()) * pc
    return emb

# 根据 SIF 计算句子向量
def SIF_embedding(word_vector, index, weight):
    # 省略根据平滑逆词频获取权重的代码
    # 加权计算
    emb = get_weighted_average(word_vector, index, weight)
    # 去除主成分
    emb = remove_pc(emb)
    return emb
```

图 4-17: SIF 代码

获取句子向量表示的代码如图4-17所示。SIF_embedding 方法的输入包括句子中的每个词语对应的词向量（word_vector），句子中词语的索引（index），句子中词语的权重（weight）。SIF_embedding 方法以平滑倒词频为权重，然后调用 get_weighted_average 方法对所有词的词向量进行加权平均，最后调用 reomve_pc 方法减去句子主成分，得到句子的向量表示。compute_pc 方法调用 TruncatedSVD 方法计算主成分，主成分即词向量各自在句向量矩阵的第一

主向量上的投影。

计算向量余弦相似度的代码如图4-18所示，针对 A，B 向量中每个元素 a，b，对 ab 相乘的结果进行求和作为分子，a^2 求和并开方后的结果与 b^2 求和并开方后的结果相乘作为分母，两者相除得到余弦距离。

```
def cos_distance(a, b):
    if len(a) != len(b):
        return None
    part_up = 0.0
    a_sq = 0.0
    b_sq = 0.0
    for a1, b1 in zip(a, b):
        part_up += a1 * b1
        a_sq += a1 ** 2
        b_sq += b1 ** 2
    part_down = math.sqrt(a_sq * b_sq)
    if part_down == 0.0:
        return 0
    else:
        return part_up / part_down
```

图 4-18: 余弦距离代码

说明文档生成的代码如图4-19所示。在重复性检测完成后，需要对结果进行说明，在前端页面进行展示，让管理员进行人工复核，避免系统自动化检测带来的问题。同时生成说明文档供众测管理员查看，下载。文档内容包括重复缺陷报告 ID，缺陷报告详细内容，与哪份缺陷报告重复，用于复盘，优化系统和对众测工人进行说明。

首先调用上文所述的相似度计算方法 similarity 进行缺陷报告相似度的计算，判断两份报告的相似度是否超过众测管理员输入的规定相似度，如果超过了，首先调用 isParent 方法判断是否具有亲缘关系，即是否某份报告是另一份报告的父报告或更高层级的祖先报告，如果具有亲缘关系则不做修改，否则根据两份缺陷报告的提交时间，调用 isEarlier 方法获得两者的提交先后关系，将较早的缺陷报告 A 作为较晚的缺陷报告 B 的置零原因，如果报告 B 已有置零原因缺陷报告 C，则比较缺陷报告 A 和 C 各自与缺陷报告 B 的相似度，以相似度较高的报告作为最终置零原因。最后每份相似度过高，较晚提交且游离于亲缘关系外的缺陷报告都会被系统认为重复，并给出理由。众测管理员针对

这些报告进行人工复核，确认无误后将重复报告的参考分置为零分，然后触发 causeDocument 方法，调用 pd.DataFrame.to_csv 生成说明文档，内容分为四列，分别为零分报告 ID，零分报告描述，重复报告 ID 和重复报告描述，便于众测管理员进行查看和人工复核，减少错判，避免带来争议。

```
def compareTwoAnswers(bug_data, similarity, biggest_similarity, a, b):
    if similarity >= biggest_similarity:
        # 是父子关系则不做改变
        if isParent(bug_data, a, b) or isParent(bug_data, b, a):
            return
        # 不是父子关系则后写的缺陷报告的分数置零
        else:
            if isEarlier(bug_data['create_time'][a],bug_data['create_time'][b]):
                bug_data['score'][b] = 0
                # 添加分数置零原因
                bug_data['set_0_id'][b] = findRealCause(bug_data['set_0_id'][b], a)
            elif isEarlier(bug_data['create_time'][b],bug_data['create_time'][a]):
                bug_data['score'][a] = 0
                bug_data['set_0_id'][a] = findRealCause(bug_data['set_0_id'][a], b)

def causeDocument(bug_data):
    result_data = pd.DataFrame()
    result_data.insert(0, 'bug_id', bug_data.pop('bug_id'))
    result_data.insert(1, 'description', bug_data.pop('description'))
    result_data.insert(2, 'set_0_id', bug_data.pop('set_0_id'))
    result_data.insert(3, 'set_0_description', bug_data.pop('set_0_description'))
    result_data.to_csv('causeDocument.csv')
```

图 4-19: 说明文档生成代码

缺陷报告重复性检测界面如图4-20所示。众测管理员输入自定义重复率，点击“报告查重”按钮，系统自动进行重复率检测，将重复率高于设定值的报告找出并展示，展示内容包括重复报告 ID，重复报告描述，置零原因报告 ID，置零原因报告描述，相似度，默认选中“是否置零”选项。管理员可以针对系统找出的重复报告进行人工复核，如果不是重复报告，取消“是否置零”即可。复核结束后，点击“重复报告置零”按钮，将重复报告的参考分置为零分。

点击可以查看两份缺陷报告的具体信息，进行对比，如图4-21所示。对比内容包括缺陷报告标题，页面，漏洞分类，严重等级，复现程度，具体描述等，通过具体信息来辅助众测管理员判断是否真的是重复报告，减少错判。

重复率： 0.95 报告查重 重复报告置零

序号	重复报告ID	重复报告描述	置零原因ID	置零原因描述	重复率	是否置零
1	10010000035557	账薄有问题	10010000035556	账薄有问题	100.00%	☑
2	10010000035566	测试	10010000035564	继续测试	100.00%	☑
3	10010000035568	test	10010000035563	test	100.00%	☑
4	10010000035569	test	10010000035563	test	100.00%	☑
5	10010000035574	账薄有问题	10010000035556	账薄有问题	100.00%	☑
6	10010000035576	饼图上有生活开支、房租、交通等选项，但是点击任何一项饼图都没有反应，也不提示任何信息，app应该在点击任何一项之后显示该项的详细开支	10010000035575	饼图上有生活开支、房租、交通等选项，但是点击任何一项饼图都没有反应，也不提示任何信息	99.31%	☑
7	10010000035599	饼图上有生活开支、房租、交通等选项，但是点击任何一项饼图都没有反应，也不提示任何信息	10010000035575	饼图上有生活开支、房租、交通等选项，但是点击任何一项饼图都没有反应，也不提示任何信息	100.00%	☑
8	10010000035621	科目存在问题	10010000035556	账薄有问题	96.91%	☑
9	10010000035703	账薄有问题	10010000035556	账薄有问题	100.00%	☑
10	10010000035706	导不出交易	10010000035602	不能交易	100.00%	☑
11	10010000035707	用户体验有问题	10010000035704	用户体验有问题	100.00%	☑
12	10010000035711	账薄有问题123156456	10010000035556	账薄有问题	100.00%	☑

图 4-20: 缺陷报告重复性检测截图

报告对比

属性	重复报告	置零原因
ID	10010000035557	10010000035556
标题	账薄有问题	账薄有问题
页面	GnuCash-账簿-管理账簿	GnuCash-账簿-管理账簿
漏洞分类	页面布局缺陷	页面布局缺陷
严重等级	严重	一般
复现程度	无规律复现	无规律复现
创建时间	2019-02-19 19:15	2019-02-19 19:13
Bug描述	账薄有问题	账薄有问题

图 4-21: 缺陷报告信息对比截图

4.5 本章小结

本章利用流程图，界面截图，逻辑解释，代码展示相结合的方式等方式对系统中知识图谱构建模块，知识图谱特征学习模块，缺陷报告推荐模块和缺陷报告重复性检测模块的具体实现进行了描述。

第五章　系统测试与实验分析

5.1　测试准备

5.1.1　测试目标

本章主要包括功能测试，性能测试和效果测试。功能测试主要针对系统的功能性需求进行测试，检查系统是否能提供相应的服务，满足功能性需求。性能测试主要针对系统的性能需求进行测试，检查系统是否满足性能要求，能在高并发下正常提供服务。效果测试主要针对系统的服务效果进行测试，检查在使用基于知识图谱的众测智能推荐技术的情况下，系统的服务能力和众测工人的满意程度是否有所提高以及提高的程度。

5.1.2　测试环境

根据 3.2.3 节的系统物理部署图，系统的测试环境包括用户机，用于运行 Chrome 浏览器，与系统进行交互；用于运行 Angular2 程序的前端服务器；用于运行部署了 Nginx 与 SpringBoot 的 Docker 的众测后端服务器；用于运行 Django 程序的推荐服务器；用于部署 Redis 与 MongoDB 集群的数据服务器。考虑到成本和运维的优势，所有服务均部署于阿里云服务器，根据服务运行的要求不同，部署在了不同配置的服务器上，具体信息如表5-1所示。

表 5-1: 系统测试环境

设备	程序	详细信息
用户机	Chrome 浏览器	MacOS 10.15
众测前端服务器	Angular2	ECS 服务器 Ubuntu 18.04
众测后端服务器	SpringBoot	ECS 服务器 4G 内存 50M 带宽
推荐服务器	Django	ECS 服务器 4G 内存 50M 带宽
数据服务器	MongoDB,Redis	ECS 服务器 4G 内存 50M 带宽

5.2 功能测试

根据第三章系统需求分析得到的功能性需求，本小节进行测试用例的设计，目标是保证基于知识图谱的众测智能推荐技术能够提供预期的功能，满足用户的业务需求，达到功能覆盖的标准。具体测试用例如下所示。

表5-2展示了填写测试报告测试用例，测试的是众测工人填写测试报告相关信息的功能，主要关注点是众测工人是否能够正常填写测试报告相关信息并保存，需要测试的具体功能包括填写测试报告，众测中途编辑测试报告，保存测试报告，填写未保存系统保留草稿。

表 5-2: 填写测试报告测试用例

用例编号	TC1
测试名称	填写测试报告测试
前置条件	用户已经得到授权和认证
正常流程	1. 点击“编辑”按钮 2. 填写测试报告名称，测试设备品牌，名称，操作系统 3. 点击“取消”按钮 4. 点击“编辑”按钮 5. 填写如步骤 2 信息 6. 点击“保存”按钮
预期结果	1. 展示测试报告信息编辑页面 2. 中途退出再次编辑，系统自动保存草稿 3. 系统保存测试报告相关信息并进行更新，展示
测试结果	通过

表5-3展示了填写测试用例测试用例，测试的是众测工人填写测试用例相关信息的功能，主要关注点是众测工人是否能够正常填写测试用例相关信息并保存，需要测试的具体功能包括填写测试用例，保存测试用例，填写未保存系统自动保留草稿。

表 5-3: 填写测试用例测试用例

用例编号	TC2
测试名称	填写测试用例测试
前置条件	用户填写了测试报告相关信息
正常流程	1. 点击“创建测试用例”按钮 2. 填写测试用例名称，前置条件，正常流程，后置条件 3. 点击“取消”按钮 4. 点击“创建测试用例”按钮 5. 填写如步骤 2 信息 6. 点击“保存”按钮
预期结果	1. 展示测试用例信息编辑页面 2. 中途退出再次编辑，系统自动保存草稿 3. 系统保存测试用例相关信息并更新测试用例列表
测试结果	通过

表 5-4: 填写缺陷报告测试用例

用例编号	TC3
测试名称	填写缺陷报告测试
前置条件	用户填写了测试报告相关信息
正常流程	1. 点击“创建 Bug”按钮 2. 选择三级页面，Bug 属性，所属用例 3. 填写缺陷报告标题和详细描述 4. 上传 Bug 截图 5. 点击“取消”按钮 6. 重复 1-4 步 7. 点击“保存”按钮
预期结果	1. 展示缺陷报告信息编辑页面 2. 中途退出再次编辑，系统自动保存草稿 3. 系统保存缺陷报告相关信息并更新缺陷报告列表
测试结果	通过

表5-4展示了填写缺陷报告测试用例，测试的是众测工人填写缺陷报告相

关信息的功能，主要关注点是众测工人是否能够正常填写缺陷报告相关信息并保存，具体信息包括三级页面，基本属性，测试用例的选择，报告标题和描述的填写，截图的上传，需要测试的具体功能包括填写缺陷报告，保存缺陷报告，填写未保存系统保留草稿。

表 5-5: 查看推荐报告测试用例

用例编号	TC4
测试名称	查看推荐报告测试
前置条件	用户正在编辑缺陷报告
正常流程	1. 进行缺陷报告的编辑 2. 查看推荐缺陷报告列表 3. 点击缺陷报告，查看具体信息
预期结果	1. 系统根据用户输入，实时更新推荐列表并展示 2. 系统展示缺陷报告具体信息，包括缺陷报告 ID，所处位置，基本属性，所属用例，标题，描述，截图，父子关系等。
测试结果	通过

表 5-6: 筛选缺陷报告测试用例

用例编号	TC5
测试名称	筛选缺陷报告测试
前置条件	用户已经得到授权和认证
正常流程	1. 选择三级页面 2. 选择缺陷报告基本属性 3. 输入关键词 4. 点击“搜索”按钮 5. 查看缺陷报告列表 6. 点击缺陷报告，查看具体信息
预期结果	1. 系统根据条件和关键词展示符合的缺陷报告列表 2. 系统展示缺陷报告具体信息
测试结果	通过

表5-5展示了查看推荐报告测试用例，测试的是众测工人填写缺陷报告时

查看系统推荐的缺陷报告相关信息的功能，主要关注点是众测工人是否能够正常获得缺陷报告推荐和查看具体信息。需要测试的具体功能包括实时推荐缺陷报告，查看缺陷报告具体信息。

表5-6展示了筛选缺陷报告测试用例，测试众测工人通过三级页面，基本属性等条件和关键词进行缺陷报告筛选的功能，主要关注点是众测工人是否能够正常进行缺陷报告筛选，查看缺陷报告具体信息。测试的具体功能包括根据条件筛选缺陷报告，根据关键词搜索缺陷报告，查看缺陷报告具体信息。

表5-7展示了审核缺陷报告测试用例，测试的是众测工人对缺陷报告进行审核的功能，主要关注点是众测工人是否能够正常对缺陷报告进行点赞点踩和取消点赞点踩。需要测试的具体功能包括对缺陷报告进行点赞，点踩，取消点赞，取消点踩。

表5-8展示了克隆缺陷报告测试用例，测试的是众测工人对缺陷报告进行克隆和优化的功能，主要关注点是众测工人是否能够正常对缺陷报告进行克隆，克隆后能否正常进行编辑优化。需要测试的具体功能包括克隆缺陷报告，继续编辑缺陷报告，提交缺陷报告，保存缺陷报告父子关系等。

表 5-7: 审核缺陷报告测试用例

用例编号	TC6
测试名称	筛选缺陷报告测试
前置条件	用户查看缺陷报告具体信息
正常流程	1. 点击“点赞”按钮 2. 再次点击“点赞”按钮取消点赞 3. 点击“点踩”按钮 4. 再次点击“点踩”按钮取消点踩
预期结果	1. 用户点赞后，“点踩”按钮消失，显示“已点赞” 2. 取消点赞后，“已点赞”消失，显示点赞点踩按钮 3. 用户点踩后，“点赞”按钮消失，显示“已点踩” 4. 取消点踩后，“已点踩”消失，显示点赞点踩按钮
测试结果	通过

表 5-8: 克隆缺陷报告测试用例

用例编号	TC7
测试名称	克隆缺陷报告测试
前置条件	用户查看缺陷报告具体信息
正常流程	1. 点击“克隆”按钮 2. 编辑缺陷报告 3. 点击“保存”按钮
预期结果	1. 系统将被克隆报告的信息填充至缺陷报告编辑界面 2. 系统保存缺陷报告相关信息并更新缺陷报告列表 3. 保存新缺陷报告为被克隆报告的子报告
测试结果	通过

表 5-9: 缺陷报告查重测试用例

用例编号	TC8
测试名称	缺陷报告查重测试
前置条件	用户已经得到认证和授权
正常流程	1. 选择要查重的众测任务 2. 输入重复率，点击“重复率检测”按钮 3. 查看检测结果 4. 对于错判的报告取消“是否置零”勾选 5. 点击“报告置零”按钮 6. 点击说明文档下载链接
预期结果	1. 展示众测任务列表 2. 展示重复率检测页面 3. 展示重复率检测结果 4. 置重复报告评分为 0 5. 生成说明文档，显示下载链接
测试结果	通过

表5-9展示了缺陷报告查重测试用例，测试的是众测管理员对缺陷报告进行重复性检测的功能，主要关注点是众测工人是否能够正常对缺陷报告进行重复性检测，自动评分，减少评审专家的工作量。需要测试的具体功能包括根据

重复率查重，对系统判定的重复报告进行人工复核，重复报告参考分置为零分，说明文档生成与下载。

由上文可以看出，本系统所有功能测试全部通过，说明本系统满足了需求分析阶段提出的功能性需求，能够提供相应的服务，满足用户的业务需求，有一定的应用价值，可以实际投入使用。

5.3 性能测试

5.3.1 测试设计

性能测试主要针对系统的接口进行高并发测试，检查在高并发情况下系统是否能正常运行并保持高速反馈，考虑到用户使用的频率和要求，需要进行测试的接口整理如表5-10。

表 5-10: 性能测试接口列表

测试接口编号	接口描述	接口路径
I1	创建测试报告	/report/create
I2	创建测试用例	/testCase/create
I3	创建缺陷报告	/bug/create
I4	获取推荐的缺陷报告	/recommend/bug
I5	获取缺陷报告具体信息	/bug/detail
I6	筛选缺陷报告	/bug/search
I7	审核缺陷报告	/bug/check

本文采用 JMeter 对接口进行测试，以筛选缺陷报告接口为例，配置 JMeter 具体步骤如下：

1）使用 JMeter 创建线程组，设置线程组中 Number of Threads=200，Ramp-Up Period=10，Loop Count=5，代表线程数量即虚拟用户数量为 2000，每个用户在 10s 内循环发送 5 次请求，共发送 1000 个请求，具体配置如图5-1所示。

2）在线程组中添加针对缺陷报告接口的 HTTP 请求，对接口进行压力测试，配置内容包括请求协议，IP，端口，请求类型，路径，编码，参数等，具体配置如图5-2所示，该图以缺陷报告推荐请求为例。

图 5-1: 线程组配置截图

图 5-2: HTTP 请求配置截图

5.3.2 测试结果

表 5-11描述了各个接口的 JMeter 测试结果汇总，可以看到，在高并发情况下，各个接口均能保持没有错误请求发生，将响应时间控制在一个较为平稳且快速的时间段内，满足系统的非功能需求。说明本系统能在高并发环境下正常服务，具有高性能，高可靠性，有较高的实用价值。

以获取推荐报告接口为例，该接口是所有接口中使用频率最高，响应时间要求最高的接口之一。如图 5-3可以看到，200 个线程在 10 秒内各自发送了 5 次缺陷报告推荐请求，共 1000 次请求，错误请求数为 0，平均响应时间为 39ms，99% 的请求在 252ms 内得到了响应。

表 5-11: 接口性能测试结果

接口编号	错误请求数量	平均响应时间	99% 响应时间
I1	0	23ms	193ms
I2	0	27ms	202ms
I3	0	28ms	214ms
I4	0	39ms	252ms
I5	0	19ms	189ms
I6	0	34ms	209ms
I7	0	22ms	195ms

Label	# 样本	平均值	中位数	90% 百分位	95% 百分位	99% 百分位	最小值	最大值	异常 %	吞吐量	接收 KB/sec	发送 KB/sec
HTTP请求	1000	39	20	76	165	252	15	345	0.00%	99.1/sec	151.54	14.61
总体	1000	39	20	76	165	252	15	345	0.00%	99.1/sec	151.54	14.61

图 5-3: 缺陷报告推荐测试结果截图

5.4 效果测试

效果测试采用对比实验进行效果的验证，使用基于知识图谱的众测智能推荐技术的众测平台，基于相似度推荐技术的众测平台和常规众测平台进行对比，检查基于知识图谱的众测智能推荐技术在众包测试任务过程中的效果，主要对比点包括报告质量，报告重复率，缺陷覆盖率，最终报告质量等。

5.4.1 测试设计

1）选用众测任务，选择经典的众测任务“咕咚翻译 APP”，该测试目标中已预先埋进多个已知的 Bug，便于后期进行整理和对比。

2）众测工人在进入测试报告编辑页面时，会随机进入三个版本中的一个，A 版本为采用了基于知识图谱的众测智能推荐技术的众包平台，B 版本为采用了相似度推荐技术的众包平台，C 版本相较于 A,B 版本，缺少了测试报告推荐，筛选，审核，克隆，查重等功能，其余报告提交，评分，整编功能均不变，目的是为了控制变量，着重考察基于知识图谱的众测智能推荐技术对于协作式众包平台的影响以及相对于相似度推荐技术的提高程度。

3）众测工人进行众测任务，任务完成后由评审专家进行评分，众测管理员进行报告整编，生成最终交付报告。

5.4.2 测试结果

表 5-12: A/B 测试执行结果

	A 组	B 组	C 组
总提交数	1002	1108	1267
重复报告数	57	237	439
重复率	5.6%	21.3%	34.6%
缺陷覆盖率	100%	86%	58%
页面全覆盖所需时间	56 分钟	85 分钟	103 分钟
平均分	7.9	7.2	5.3

在规定的三小时众测任务时间内，共有 300 名众测工人参与本次众测任务，其中 A 版本 100 人，B 版本 100 人，C 版本 100 人，测试结果如表5-12所示。第一章提出了众包测试平台目前面临的四个问题，重复报告数量高，测试页面覆盖率低，缺陷报告质量差，众包工人积极性差。

从表中可以看出，基于知识图谱的众测智能推荐技术相对于基于相似度的报告推荐和常规众测平台，减少了重复报告数，降低了重复率；提高了缺陷覆盖率和测试页面全覆盖所需的时间。

缺陷报告质量度量方面，采用缺陷报告平均分进行度量，在众包测试任务结束后，评审专家针对每个缺陷报告进行打分，可以看到，A 组的平均分最高，人均的高质量报告（6 分及以上）数也最高，在一定程度上反映了缺陷报告的质量有所提高。

工人积极性方面，虽然 A 组相对于其他两个版本，总提交数有所下降，工人积极性似乎有所下降，但可以认为是报告推荐起到了一定效果，减少了一部分重复报告的提交，减去重复报告数后，A 组的有效报告数实际上是最多的，一定程度上说明了工人积极性有所提高。

基于知识图谱的众测智能推荐技术相比基于相似度的推荐技术，推荐结果更为多样，以用户填写属性为“标题：登录失败，描述：1. 输入账号密码，2. 点击“登录”按钮，3. 显示登录失败”的缺陷报告为例，基于相似度的推荐技术会

重点推荐标题或描述中含有“登录失败”等的缺陷报告，基于知识图谱的推荐技术在这之上，会根据用户对于历史报告的关注数据，发现用户的喜好，推荐一些与“账号”，“密码”相关的缺陷报告，如“密码重设失败”，“账号昵称无法修改”等；可解释性方面，两者的推荐理由都较坚实，一个是基于知识图谱的路径推理和用户兴趣预测，一个是基于报告相似度，但基于知识图谱的推荐技术考虑到用户的历史数据而非单纯的正在编辑数据，可解释性更强；同时，为了更直观地度量两者的精确性，采用精确率和人均点击次数进行比较。

精确率为用户点击推荐报告占所有推荐报告的占比，定义如公式5-1所示，其中 TP 代表用户点击的报告数，FP 代表用户未点击的报告数，两者之和为系统推荐的报告总数。

$$pec = \frac{TP}{TP + FP} \tag{5-1}$$

人均点击次数即每个众包工人点击推荐报告的平均值，定义如公式5-2所示，其中C代表用户点击总数，uv代表用户数。

$$c = \frac{C}{uv} \tag{5-2}$$

精确率和人均点击次数示意图如图5-4所示，横坐标为众测任务时间，单位为分钟，共三小时，因为缺陷报告推荐的频率较高，所以精确率两者均较低，人均点击次数随着时间均有所上升，但无论是精确率还是人均点击次数，基于知识图谱的众测智能推荐技术的表现都更为优秀，说明众包工人更愿意点开基于知识图谱的众测智能推荐技术推荐的缺陷报告，推荐效果更好。

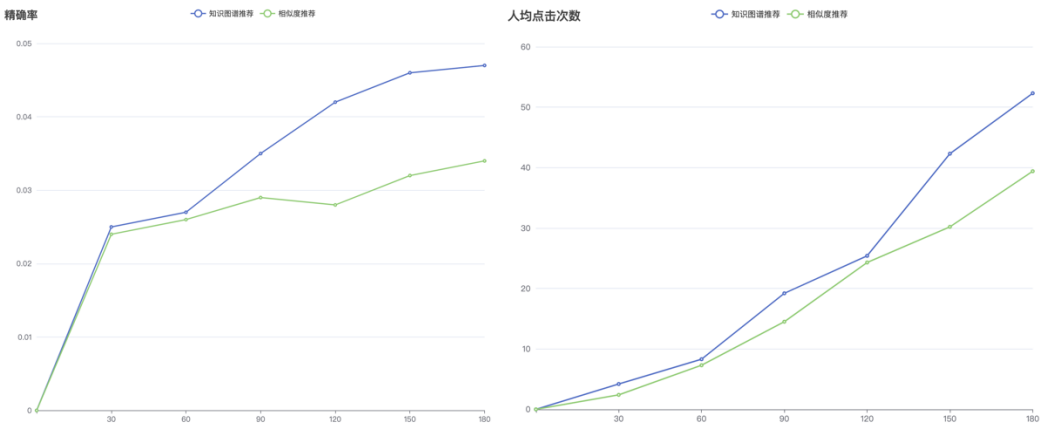

图 5-4: 精确率和人均点击次数

总结以上，基于知识图谱的众测智能推荐技术在一定程度上优化了众包测试平台目前面临的一些问题，满足了用户的业务需求，更好地实现众包测试的目标，为给任务发布者提交满意的报告做出了贡献。

5.5 本章小结

本章对系统的功能和性能进行了充分的测试，测试结果表明系统能够正常提供必要的功能，满足用户的业务需求，在高并发和异常情况下也可正常运行，满足用户对于高性能和可靠性的要求。同时通过效果测试说明了基于知识图谱的众测智能推荐技术对于协作式众包测试平台的多方面影响和提高，包括降低重复报告率，提高报告质量，增强页面覆盖率，提高众包工人积极性，具有较高的实用价值。

第六章　总结与展望

6.1　总结

众包测试能够提高测试效率，降低测试成本，因此受到了越来越多企业和个人的关注。但众包测试在工业界落地的过程中，出现了重复报告数量多，缺陷报告质量差，测试页面覆盖率低，众包工人积极性差等诸多问题，本文针对以上问题，结合协作式众包平台的现状，提出了基于知识图谱的众测智能推荐技术，开发了众测智能推荐系统。

众包工人在进行众测任务的过程中，会不断被动接受系统推荐的缺陷报告，工人也可以主动通过条件和关键词进行缺陷报告查找和筛选，通过查看他人的缺陷报告，众包工人可以了解其他人的进度和成果，不对已经发现的 Bug 进行挖掘，从源头减少重复报告的产生，也会产生一定的竞争心理，提高工人的积极性。同时，众包工人会主动前往尚未发现 Bug 的页面进行测试和挖掘，主观上希望能发现新的 Bug，客观上提高了测试页面的覆盖率。众包工人查看缺陷报告后，可以进行审核，表达自己对于该缺陷报告的评价，间接影响该缺陷报告的优先级和最终交付报告的质量。众包工人也可以对缺陷报告进行克隆，在他人提交报告的基础上进行优化，发挥群体智能的力量，提高对于一个 Bug 的描述质量。众测任务结束后，众测管理员在评审开始前，可以进行缺陷报告的重复性检测，系统发现重复报告并自动置零分，减少评审和整编工作量，提高报告质量。

本文首先介绍了项目的背景和意义，包括众包测试和知识图谱推荐技术的研究现状，受到现有工作的启发，提出了基于知识图谱的众测智能推荐技术，将知识图谱与推荐系统相结合。其次，对于本文使用的科学理论，技术框架和开源工具进行了介绍，详细阐述了选择相应工具的背景及理由。接着，对系统进行了需求分析，得到了系统用例，之后根据分析结果对系统进行了架构设计，模块划分和实体类设计，采用 4+1 视图从不同角度对系统进行了描述，然后对系统使用的推荐算法和重复率检测算法进行了设计与描述。接着使用流程图，类图，时序图，关键代码和截图相结合的方式对技术的详细设计与系统的

具体实现进行了阐述。最后对系统进行了充分的测试，保证了系统的可用性，可靠性和算法效果的实际提升。

目前，基于知识图谱的众测智能推荐技术已经在真实众包测试平台上投入使用，服务于众多企业，高校和个人，已为几十场众包测试任务提供了推荐和查重服务，减少了重复报告，提高了测试页面覆盖率，优化了报告质量，提高了众包工人的积极性，取得了良好的实际效果，为企业创造了一定的价值。

6.2 展望

基于知识图谱的众测智能推荐技术虽然已投入使用并取得了一定的效果，但系统实现目前仍有许多需要改进的地方，具体如下所示：

1）未考虑图像因素。知识图谱中的实体只考虑了文本描述的关键词和一些其它相关属性特征，未将众包工人上传的 Bug 截图考虑在内，无论是推荐还是查重模型均可进行改进，考虑图像因素，提高准确率。

2）无法进行端到端的训练。依次学习的优点在于知识图谱特征学习模块和推荐模块相互独立，知识图谱模块不需要与推荐模块一起更新，但也正因为独立，系统无法做到端到端的训练和优化。后续可以考虑使用联合学习和交替学习进行模型更新，对比效果。

3）推荐优化机制不完善。本文对于推荐结果和用户之后的行为进行了记录，但并未进行分析，对推荐系统进行优化。后续可以考虑根据用户对于推荐报告的行为进行推荐系统的优化，提高推荐的效率和准确率。

综上，本文提出的基于知识图谱的众测智能推荐技术仍有许多可以继续提升和优化的部分。可以通过考虑图像特征提高推荐和查重模型的准确性；通过采用联合学习，交替学习等方式进行端到端的训练，提高推荐效果；通过对推荐结果和用户行为进行记录分析，优化推荐系统，提高推荐效果。

致　　谢

行文至此，意味着我的研究生生活即将落幕，回首在南京大学的求学光阴，无论是在仙林校区度过的前两年，还是在鼓楼校区度过的后四年，都是我青春的美好回忆，始于 2015 年金秋，终于 2021 年盛夏，万般不舍，终有一别。

桃李不言，下自成蹊，首先要感谢我的导师陈振宇教授，在硕士期间给予我学业和生活上的帮助，还要感谢冯洋老师，本文从选题到设计到最后的多次修改和定稿，都离不开两位老师的指导和帮助，在此表达最诚挚的感谢。

其次，要感谢实验室的所有同学，在硕士期间我们共同进行众测相关系统的开发，合作无间，在毕业设计项目开发和论文撰写的过程中也互相讨论，帮助我完成项目和毕业论文，在他们身上我也学习到很多优秀的品质。

然后，我要感谢南京大学软件学院对我的培养，历时六年，我从一个不懂计算机技术的人成为了一个合格的软件开发工程师，感谢软件学院所有老师的谆谆教诲，我会将“嚼得菜根，做得大事”牢记于心，在日后的职业生涯中，不负南大之盛名。

我还要感谢我的父母，在我本科和硕士期间负担了我的学费和生活费，感谢他们对我二十余年的培养，在读书阶段对我的鼓励和支持，在遇到困难时对我的关心和理解。

感谢所有参与毕业设计评审的老师和专家们，你们辛苦了。在以后的生活工作中，我一定会更加努力，成为一个更优秀的人。

须知少年凌云志，曾许人间第一流。感谢在过去十八年求学生涯中不曾放弃的自己，如今即将走向社会，希望未来的自己，不要被遇到的困难磨平棱角，不要忘记自己最初的模样，不忘初心，方得始终。

参考文献

[1] HOWE J. The rise of crowdsourcing[J]. Wired magazine, 2006, 14(6): 1–4.

[2] SHEN H, LI Z, LIU J, et al. Knowledge sharing in the online social network of yahoo! answers and its implications[J]. IEEE transactions on computers, 2014, 64(6): 1715–1728.

[3] 张志强, 逄居升, 谢晓芹, et al. 众包质量控制策略及评估算法研究 [J]. 计算机学报, 2013, 36(8): 1636–1649.

[4] ZOGAJ S, BRETSCHNEIDER U, LEIMEISTER J M. Managing crowdsourced software testing: a case study based insight on the challenges of a crowdsourcing intermediary[J]. Journal of Business Economics, 2014, 84(3): 375–405.

[5] 章晓芳, 冯洋, 刘峘, et al. 众包软件测试技术研究进展 [J]. Journal of Software, 2018, 29(1).

[6] CHENG P, LIAN X, CHEN L, et al. Task assignment on multi-skill oriented spatial crowdsourcing[J]. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(8): 2201–2215.

[7] 冯剑红, 李国良, 冯建华. 众包技术研究综述 [J]. 计算机学报, 2015, 38(9): 1713–1726.

[8] ALLAHBAKHSH M, BENATALLAH B, IGNJATOVIC A, et al. Quality control in crowdsourcing systems: Issues and directions[J]. IEEE Internet Computing, 2013, 17(2): 76–81.

[9] ZHANG X, CHEN Z, FANG C, et al. Guiding the crowds for Android testing[C] // Proceedings of the 38th International Conference on Software Engineering Companion. 2016: 752–753.

[10] LEASE M. On quality control and machine learning in crowdsourcing.[J]. Human Computation, 2011, 11(11).

[11] BOGERS M, AFUAH A, BASTIAN B. Users as innovators: a review, critique, and future research directions[J]. Journal of management, 2010, 36(4): 857 – 875.

[12] GARDLO B, EGGER S, SEUFERT M, et al. Crowdsourcing 2.0: Enhancing execution speed and reliability of web-based QoE testing[C] // 2014 IEEE International Conference on Communications (ICC). 2014: 1070 – 1075.

[13] LIU D, BIAS R G, LEASE M, et al. Crowdsourcing for usability testing[J]. Proceedings of the American Society for Information Science and Technology, 2012, 49(1): 1 – 10.

[14] DOLSTRA E, VLIEGENDHART R, POUWELSE J. Crowdsourcing gui tests[C] // 2013 IEEE Sixth International Conference on Software Testing, Verification and Validation. 2013: 332 – 341.

[15] KOMAROV S, REINECKE K, GAJOS K Z. Crowdsourcing performance evaluations of user interfaces[C] // Proceedings of the SIGCHI conference on human factors in computing systems. 2013: 207 – 216.

[16] GÓMEZ M, ROUVOY R, ADAMS B, et al. Reproducing context-sensitive crashes of mobile apps using crowdsourced monitoring[C] // 2016 IEEE/ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft). 2016: 88 – 99.

[17] CHEN F, KIM S. Crowd debugging[C] // Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. 2015: 320 – 332.

[18] BLANCO R, HALPIN H, HERZIG D M, et al. Repeatable and reliable search system evaluation using crowdsourcing[C] // Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval. 2011: 923 – 932.

[19] 陈英奇, 赵宇翔, 朱庆华. 科研众包视角下公众科学项目的任务匹配模型研究 [J]. 图书情报知识, 2018(3): 5 – 15.

[20] PUJARA J, MIAO H, GETOOR L, et al. Knowledge graph identification[C] // International Semantic Web Conference. 2013 : 542 – 557.

[21] 徐增林, 盛泳潘, 贺丽荣, et al. 知识图谱技术综述 [J], 2016.

[22] BOLLACKER K, COOK R, TUFTS P. Freebase: A shared database of structured general human knowledge[C] // AAAI : Vol 7. 2007 : 1962 – 1963.

[23] BIZER C, LEHMANN J, KOBILAROV G, et al. DBpedia-A crystallization point for the Web of Data[J]. Journal of web semantics, 2009, 7(3) : 154 – 165.

[24] SUCHANEK F M, KASNECI G, WEIKUM G. Yago: A large ontology from wikipedia and wordnet[J]. Journal of Web Semantics, 2008, 6(3) : 203 – 217.

[25] 黄立威, 江碧涛, 吕守业, et al. 基于深度学习的推荐系统研究综述 [J]. 计算机学报, 2018, 41(7) : 1619 – 1647.

[26] 常亮, 张伟涛, 古天龙, et al. 知识图谱的推荐系统综述 [J]. 智能系统学报, 2019, 14(2) : 207 – 216.

[27] LASSILA O, SWICK R R, OTHERS. Resource description framework (RDF) model and syntax specification[J], 1998.

[28] WEBBER J. A programmatic introduction to neo4j[C] // Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity. 2012 : 217 – 218.

[29] QIN C, ZHU H, ZHUANG F, et al. 基于知识图谱的推荐系统研究综述 [J]. Scientia Sinica Informationis, 2020, 50(7) : 937 – 956.

[30] BORDES A, USUNIER N, GARCIA-DURAN A, et al. Translating embeddings for modeling multi-relational data[C] // Neural Information Processing Systems (NIPS). 2013 : 1 – 9.

[31] WANG Z, ZHANG J, FENG J, et al. Knowledge graph embedding by translating on hyperplanes[C] // Proceedings of the AAAI Conference on Artificial Intelligence : Vol 28. 2014.

[32] LIN Y, LIU Z, SUN M, et al. Learning entity and relation embeddings for knowledge graph completion[C] // Proceedings of the AAAI Conference on Artificial Intelligence : Vol 29. 2015.

[33] JI G, HE S, XU L, et al. Knowledge graph embedding via dynamic mapping matrix[C] // Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers). 2015 : 687 – 696.

[34] YANG D, GUO Z, WANG Z, et al. A knowledge-enhanced deep recommendation framework incorporating gan-based models[C] // 2018 IEEE International Conference on Data Mining (ICDM). 2018 : 1368 – 1373.

[35] DONG Y, CHAWLA N V, SWAMI A. metapath2vec: Scalable representation learning for heterogeneous networks[C] // Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. 2017 : 135 – 144.

[36] PALUMBO E, RIZZO G, TRONCY R. Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation[C] // Proceedings of the eleventh ACM conference on recommender systems. 2017 : 32 – 36.

[37] GROVER A, LESKOVEC J. node2vec: Scalable feature learning for networks[C] // Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016 : 855 – 864.

[38] WANG H, ZHANG F, XIE X, et al. DKN: Deep knowledge-aware network for news recommendation[C] // Proceedings of the 2018 world wide web conference. 2018 : 1835 – 1844.

[39] ZHANG F, YUAN N J, LIAN D, et al. Collaborative knowledge base embedding for recommender systems[C] // Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016 : 353 – 362.

[40] WANG H, ZHANG F, WANG J, et al. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems[C] // Proceedings of the 27th

ACM International Conference on Information and Knowledge Management. 2018 : 417 – 426.

[41] WANG H, ZHANG F, ZHAO M, et al. Multi-task feature learning for knowledge graph enhanced recommendation[C] // The World Wide Web Conference. 2019 : 2000 – 2010.

[42] NII M, TUCHIDA Y, IWAMOTO T, et al. Nursing-care text evaluation using word vector representations realized by word2vec[C] // 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). 2016 : 2165 – 2169.

[43] LAZARIDOU A, PHAM N T, BARONI M. Combining language and vision with a multimodal skip-gram model[J]. arXiv preprint arXiv:1501.02598, 2015.

[44] PENG H, LI J, SONG Y, et al. Incrementally learning the hierarchical softmax function for neural language models[C] // Proceedings of the AAAI Conference on Artificial Intelligence : Vol 31. 2017.

[45] GOLDBERG Y, LEVY O. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method[J]. arXiv preprint arXiv:1402.3722, 2014.

[46] AIZAWA A. An information-theoretic perspective of tf–idf measures[J]. Information Processing & Management, 2003, 39(1) : 45 – 65.

[47] ARORA S, LIANG Y, MA T. A simple but tough-to-beat baseline for sentence embeddings[J], 2016.

[48] KUSNER M, SUN Y, KOLKIN N, et al. From word embeddings to document distances[C] // International conference on machine learning. 2015 : 957 – 966.

[49] Bernstein D. Containers and Cloud: From LXC to Docker to Kubernetes[J/OL]. IEEE Cloud Computing, 2014, 1(3) : 81 – 84.
http://dx.doi.org/10.1109/MCC.2014.51.

[50] Goldberg R P. Survey of virtual machine research[J/OL]. Computer, 1974, 7(6) : 34 – 45.
http://dx.doi.org/10.1109/MC.1974.6323581.

[51] 刘熙, 胡志勇. 基于 Docker 容器的 Web 集群设计与实现 [J]. 收藏, 2016, 8.

[52] BOETTIGER C. An introduction to Docker for reproducible research[J]. ACM SIGOPS Operating Systems Review, 2015, 49(1) : 71 – 79.

[53] Wang Y, Zhang L, Tan J, et al. HydraDB: a resilient RDMA-driven key-value middleware for in-memory cluster computing[C/OL] // SC '15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. 2015 : 1 – 11.
http://dx.doi.org/10.1145/2807591.2807614.

[54] Cao W, Sahin S, Liu L, et al. Evaluation and Analysis of In-Memory Key-Value Systems[C/OL] // 2016 IEEE International Congress on Big Data (BigData Congress). 2016 : 26 – 33.
http://dx.doi.org/10.1109/BigDataCongress.2016.13.

[55] KRUCHTEN P B. The 4+ 1 view model of architecture[J]. IEEE software, 1995, 12(6) : 42 – 50.

简历与科研成果

基本信息

徐佳炜，男，汉族，1997 年 7 月出生，江苏省南通人。

教育背景

2019 年 9 月 — 2021 年 6 月 南京大学软件学院 硕士

2015 年 9 月 — 2019 年 6 月 南京大学软件学院 本科