

基于用户评论的代码质量识别与分析

徐海燕 姜 瑛

云南省计算机技术应用重点实验室 昆明 650500

昆明理工大学信息工程与自动化学院 昆明 650500

(1159381349@qq.com)



摘 要 随着 IT 社区和代码托管平台的发展,针对代码的用户评论数量急剧增加。用户在使用代码后给出的评论中包含丰富的静态和动态代码质量信息,对其进行提取与分析将有助于开发者了解用户关注的代码质量信息,以有针对性地提升代码质量,还有助于用户选择满足要求的代码。为此,文中提出了包含静态特性和动态特性的代码质量模型,以及识别并分析用户评论中代码质量信息的方法。首先,根据评价对象和评价句型规则识别出具有代码质量的用户评论;然后,应用评价对象和评价观点抽取代码质量属性表现;最后,通过分析代码质量属性表现和情感倾向给出代码静态和动态质量的相关结果。实验结果表明,所提方法能够有效地分析用户评论中的代码质量信息。

关键词: 代码质量;用户评论;评价对象;评价观点;评价句型;质量属性表现

中图法分类号 TP311

Code Quality Recognition and Analysis Based on User's Comments

XU Hai-yan and JIANG Ying

Computer Technology Application Key Lab of Yunnan Province, Kunming 650500, China

Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China

Abstract With the development of IT community and code hosting platforms, the number of user's comment about the code increasings dramatically. The comments given by users after using the code contain plenty of static and dynamic code quality information. The extraction and analysis of code quality information will help developers to understand the code quality information concerned by users and improve the quality of code. It is also helpful to users choose the code to meet the requirements. To this end, this paper proposed a code quality model including static and dynamic characteristics and a method to identify and analyze the code quality information in user's comments. Firstly, the users' comments with code quality are identified according to the evaluation objects and the evaluation sentence pattern rules. Secondly, the representations of the code quality attribute are extracted by using the evaluation objects and opinions. Finally, the related results of static and dynamic code quality are gained after analyzing the quality attributes representations and emotional tendency of code in user's comments. The experimental results show that the proposed method can effectively analyze the code quality information in user's comments.

Keywords Code quality, User's comments, Evaluation object, Evaluation opinion, Evaluation pattern, Quality Attribute Representation

1 引言

近年来,IT 社区以及代码托管平台中的信息已经成为开发者编程中的重要参考点。在软件产品中, Panago 等指出由于软件开发者和用户互不认识,用户使用反馈(如建议意见和系统异常等)对开发者尤为重要^[1]。Lu 等认为针对产品的在线用户评论可以及时地体现用户关注的产品属性、同类型其

他产品的优势、自身产品的劣势,以及下一版本需要改进的地方^[2]。与软件产品类似,代码也可以通过用户评论建立开发者与用户之间的联系,并挖掘用户评论中用户关注的代码质量信息。针对代码的用户评论包含丰富的代码质量信息,其不仅能够为新用户提供选择参考,而且可以为代码开发者提供反馈信息,帮助开发者提高代码质量。

对代码进行评论的用户群体较多,专业水平相差较大。

到稿日期:2019-11-15 返修日期:2020-01-16 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。

基金项目:国家重点研发计划项目(2018YFB1003904);国家自然科学基金(61462049,61063006,60703116);云南省应用基础研究计划重点项目(2017FA033)

This work was supported by the National Key Research and Development Program of China (2018YFB1003904), National Natural Science Foundation of China (61462049,61063006,60703116) and Key Project of Yunnan Applied Basic Research (2017FA033).

通信作者:姜瑛(jy_910@163.com)

不同于普通的用户评论,代码评论具有评论领域性强、评论粒度差别大、评论文本复杂度高、文本短小、不规范的特点,导致用户评论中代码质量信息难以被识别和分析。因此,分析针对代码的用户评论,并从中获取与质量相关的信息,将有助于软件开发人员判断和提高代码质量。

本文的主要贡献有:1)针对代码的用户评论具有强领域性的特点和用户对不同质量属性评论的规律,基于包含静态特性和动态特性的代码质量模型,根据评价对象和评价句型识别出具有代码质量的用户评论;2)基于识别出的具有代码质量属性的用户评论,应用评价对象和评价观点抽取代码质量属性表现,并分析代码质量属性表现和情感倾向,给出代码静态和动态质量的相关结果。

2 相关工作

针对代码的质量度量,国内外学者开展了相关研究。Venkatasubramanyam 等使用 EMISQ (Internal Software Quality Evaluation Method)和 DAISQ(Dynamic Analysis For Internal Software Quality)对软件代码质量进行静态和动态分析,并对分析结果进行详细的对比说明^[3]。Huang 等对 Java 代码质量分析进行研究,将代码规模、规范性、可维护性、可扩展性和潜在危险定义为代码的静态质量特性,使用静态分析方法划分代码质量特性和度量元,然后使用静态测试工具获取的度量元数据来评价 Java 代码质量^[4]。Liu 等对软件中的克隆代码进行研究,使用贝叶斯网络训练数据得到克隆代码的质量预测模型,并应用 EMISQ 来评估预测结果^[5]。Xu 等对含代码的 IT 社区中问题的答案进行研究,分析问题和答案中源码的相似度,同时基于度量元分析答案代码的质量^[6]。当前针对代码质量的研究主要是根据代码测试结果来度量代码质量,一旦无法执行测试,将难以度量代码质量。

随着互联网的发展,在线用户评论与问答系统成为用户了解产品质量相关信息的一个有效途径。Poche 等基于 YouTube 编码视频中的用户评论进行分析,自动对用户问题和关注点进行总结和分类,结果表明支持向量机可以检测出有用的用户评论,Sumbasic(一种基于频率提取冗余空值的摘要技术)能够充分捕捉用户评论中的主要关注点^[7]。Zhang 等针对推荐系统中的用户评论,采用深度情感分析和多视图协调融合推荐方法,实现了个性化的用户物品推荐^[8]。Feng 等将社交影响量化因素和用户评论文本特征的有机融合注入推荐模型,实现了基于用户评论卷积网络模型的汽车推荐模型^[9]。Zhang 等基于双向分层语义神经网络模型捕获评论中词语所表达的情感,并利用评论、新闻和用户投票等多种信息源对在线新闻评论的情绪进行标注,有效地分析了在线新闻用户评论的情绪^[10]。Chen 等通过基于支持向量机与主题模型结合的评论分析方法,分析了 360 手机助手中心评分高(今日头条)和评分低(360 云盘)的两个应用中的用户差评数据,挖掘出用户针对应用差评的反馈内容,使开发者更加快捷、方便地理解用户的反馈^[11]。在软件开发人员聚集的论坛或社区中,代码使用者会针对代码的应用情况给出相应的评论,部分评论体现了代码质量的相关信息,但现有研究对其关注较少。

通过分析上述研究可以发现:

(1)大多数现有研究主要根据测试来度量代码的质量,对于无法进行测试的用户,难以应用测试方法度量代码质量。例如,文献[3]应用 EMISQ 动态分析系统和 DAISQ 动态分析系统分析代码质量;文献[4]使用静态测试工具获取度量元数据;文献[5]使用 EMISQ 评估预测代码质量。对于无法测试的代码,这些方法将难以度量其质量。

(2)大部分研究主要针对代码的静态质量。例如,文献[3]将代码规模、规范性等定义为代码静态质量特性,并根据静态质量特性度量 Java 代码质量;文献[6]分析问题和答案中源码的相似度,然后基于度量元度量答案源码的静态质量;文献[12]通过调查学生、教育工作者和专业开发人员对代码质量的认知发现,可读性与结构主要作为代码质量的描述。然而,代码在实际运行过程中产生了很多动态质量特性,这些特性在大多数代码质量度量中没有体现。

(3)当前针对用户评论的研究主要是挖掘具有一定产品信息的用户评论,而未做进一步的分析。例如,文献[11]只挖掘出用户针对应用差评的反馈内容,文献[13]挖掘出 3 种体现使用反馈的 APP 用户评论。由于其没有进一步分析挖掘出的用户评论,导致评论中包含的与产品质量相关的、有价值的信息被忽略。

鉴于针对代码的用户评论中同时包含丰富的静态和动态代码质量信息,对于无法执行测试的代码使用者而言,用户评论是其获得代码质量信息的主要途径,本文从用户角度出发,针对代码的静态和动态质量,提出了一种基于用户评论的代码质量识别与分析方法。

3 代码质量模型

代码质量分析的基础是代码质量的定义与识别,代码在编译运行前就具有的质量信息一般作为代码的静态质量,如代码规模、代码结构、注释率、编写规范和逻辑设计等。而在针对代码的用户评论不仅具有静态质量信息,还包含大量代码编译运行过程中体现出的质量信息,例如代码是否可编译、运行的性能、潜在危险等。对于代码编译运行中体现的这类质量信息,本文将将其划分为代码的动态质量。

为了度量代码的静态质量和动态质量,基于 McCall 软件质量度量模型^[14],本文依据正交原理划分质量属性,各质量属性之间不重叠,从而将静态质量划分为可读性、编程规范性、可重用性和逻辑性。其中,可读性包括代码的简洁性和可理解性;编程规范性包括代码的编写规范和结构规范;可重用性包括代码的一般性和完整性;逻辑性包括代码的算术逻辑、控制逻辑和设计开发逻辑。动态质量包括功能性、可测试性、可维护性、运行性能、可靠性、可移植性和易用性。其中,功能性包括代码的正确性和有效性;可测试性包括代码的可编译性和可运行性;运行性能包括代码运行的时间效率、存储性能和实现效果;可靠性包括健壮性、容错性和安全性。

此外,针对代码的用户评论具有很强的领域特性,评论中的大量领域性词语可以用来标识某种质量属性。例如,“注释”可以用来识别可读性,“报错”可以用来识别可靠性。本文通过分析大量用户评论发现,名词通常与代码静态质量具有强相关性,动词一般与代码动态质量具有强相关性。基于这种规律,本文搜集了与 11 个质量属性具有强相关性的 55 个

领域性词语作为基于评价对象识别具有质量属性用户评论的特征词库和基于评价句型的质量属性识别规则的约束。

综上,结合用户评论和相关研究,本文提出了一个代码质量模型,如表 1 所列。

表 1 代码质量模型

Table 1 Code quality model

类型	质量属性	含义	特征词示例
静态质量	可读性	代码以简单易理解的方式实现其功能,并容易为用户所分析的性质	注释、重复
	编程规范性	代码符合编程语言的命名和结构规范,以及设计开发中的规范性要求	缩进、格式
	可重用性	代码所提供的功能具有应用范围广、代码之间依赖性小、代码内依赖性大的性质	可用、帮助
	逻辑性	代码的算术计算、控制流程和设计开发逻辑运行正确	溢出、越界
动态质量	功能性	代码满足用户的功能,且功能全部或部分有效可用	用途、功能
	可测试性	代码在给定的测试环境下,可支持编译测试的程度	编译、测试
	可维护性	代码在新的功能改善和扩充时,可被修改的容易程度	过时、失效
	运行性能	代码在其运行过程中表现的性能,包括时间复杂度、空间复杂度和实现效果	运行、效果
	可靠性	代码在其他环境或操作下能否保证运行正确的性质	报错、bug
	可移植性	在代码运行条件发生改变时无需修改就可运行,即一次编辑到处运行的性质	兼容、版本
	易用性	代码简洁明了,不冗余,用户易于操作	好用、方便

表 1 中的代码质量模型为识别具有代码质量的用户评论和分析用户评论中代码质量的相关信息提供了依据。

4 识别具有代码质量的用户评论

针对代码的用户评论不仅包含具有代码质量的用户评论,而且包含了大量对代码质量分析没有价值的无关评论。为了提高代码质量分析的效率和相关结果的有效性,在代码质量分析之前需要预处理用户评论,抽取评论中与代码质量直接相关的评论内容,然后,据此识别含有代码质量的用户评论。

4.1 用户评论预处理

不同于其他产品的用户评论,针对代码的用户评论具有专业性、成分复杂、口语化、文本短等特点。此外,评论中还包含大量的代码片段和程序运行中开发平台的提示信息,在识别具有代码质量属性的用户评论这类信息时难以被识别。因此,为了有效地提取用户评论的对象,量化用户对代码质量的情感倾向,需要先对用户评论进行预处理。

用户评论预处理首先需要对用户评论去除停用词,将用户评论中包含的大量代码片段和程序运行出错的提示信息定义为长度超过 15 的英文字符串,并对其进行删除处理;然后,采用 NLPIR-ICTCLAS 分词工具对用户评论进行分词与词性标注。相关示例如表 2 所列。

表 2 用户评论预处理示例

Table 2 Examples of user’s comment preprocessing

原始评论	去停用词	预处理后评论
博主,你的代码我编译不成功	代码编译不成功	代码/n 编译/v 不/d 成功/a
老哥我接收到的数据乱码怎么解决	接收数据乱码怎么解决	接收/v 数据/n 乱/d 码/v 怎/ryv 么/nr1 解决/v
我的总是发布不了 org.apache.cxf.interceptor.Fault;Could not start Jetty server on port 80;Cannot assign requested address;bind,换了端口也不行,什么原因	总是发布不了 80 换了端口不行 什么原因	总是/d 发布/v 不/d 了/y 80/m 换/v 了/u 端口/n 不行/a 什/ng 么/nr1 原因/n

4.2 具有质量属性的用户评论识别

经过预处理后,不具有代码质量信息的用户评论仍然大量存在,其将影响代码质量相关信息的分析。因此,在分析代

码质量之前,先在针对代码的用户评论中抽取评价对象与评价观点,以获取与代码质量属性直接相关的评论内容;然后,基于评价对象与评价句型识别具有质量属性的用户评论。

4.2.1 抽取评价对象与评价观点

在针对代码的用户评论中,评价对象表示用户评论包含的代码质量信息主体,多为名词或者名词短语。通过分析大量评论发现,用户评论中包含大量用户对代码的动态质量行为的评价,如用户评论“编译/v 出错/v”中的评价对象为“编译/v”。因此,本文主要针对名词、动词及其组合形式建立词性组合规则以抽取评价对象,抽取规则及示例如表 3 所列,其中,n 为名词,v 为动词,vi 为不及物动词,vn 为动名词。

Table 3 Extraction rules and examples of evaluation object			
规则	优先级	示例	评价对象
n+n	1	代码/n 格式/n 能/v 不能/v 整理/v 一/m 下/f	代码/n 格式/n
n+v	2	数据/n 合并/v 怎/ryv 么/nr1 合并/v 呢/y	数据/n 合并/v
v+n	3	调试/v 结果/n 有/v 问题/n	调试/v 结果/n
v+v	4	退出/v 下载/v 操作/n 失败/vi	退出/v 下载/v
n	5	有/v DEMO /n 就/d 好/a	DEMO/n
vn/v/vi	6	编译/v 失败/vi	编译/v

在针对代码的用户评论中,评价观点表示用户评论包含的代码质量信息主体的情感倾向。形容词或动词通常作为判别句子中情感倾向的依据;副词作为形容词及动词的修饰词,起到增强情感的作用;否定副词在修饰表示评价观点形容词和动词时,起到否定的作用。因此,本文主要针对形容词、动词、否定副词及其组合形式建立评价观点词性规则以抽取评价观点。抽取规则及示例如表 4 所列,其中,a 为形容词,d 为副词,ad 为形容词性副词,vg 为动词性语素。

对评价观点的情感倾向进行量化表示,可以快速地对文本的情感倾向进行定量分析。本文在抽取评价观点之后,应用汉语情感极值表对评价观点情感倾向进行计算,得到评价观点的情感极值。例如,评论“注释/n 好/a”的评价观点为“好/a”,“好”在汉语情感词典中的情感极值为 1.0385,则评论“注释/n 好/a”的评价观点的情感极值为 1.0385。

表 4 评价观点抽取规则及示例

规则	优先级	示例	评价观点
a+a	1	写/v 简单/a 详细/a	简单/a 详细/a
d+a	2	数据/n 类型/n 不/d 一样/a	不/d 一样/a
d+d+a	3	阈值/n 设置/v 不/d 大/d 理想/a	不/d 大/d 理想/a
d+v	4	不/d 支持/v locations/n	不/d 支持/v
v+u	5	代码/n 配置/vn 少/v 了/u 依赖/v	少/v 了/u
v+v	6	感谢/v 分享/v	感谢/v 分享/v
a+n	7	谢/v 大/a 神/n 不/容/v 分享/v	大/a 神/n
n+v	8	膜/n 拜/v 源码/n	膜/n 拜/v
a/ad	9	写/v 清晰/a 明/vg 了/u 很/d 详细/ad	清晰/a 详细/ad
vg/v/vi	10	赞/vg 学习/v 了/y 编译/v 失败/vi	赞/vg 学习/v 失败/vi

4.2.2 基于评价对象与评价句型识别具有质量属性的用户评论

通过抽取评价对象与评价观点,获取用户评论中与代码质量属性直接相关的内容。为了筛选具有代码质量属性的用户评论,本文依据表 1 中的代码质量模型,应用基于评价句型与评价对象的质量属性识别规则来识别具有代码质量属性的用户评论。

现有的针对用户评论挖掘的相关研究通常忽略了用户评

论句型在数据挖掘中的规律。本文通过分析大量用户评论发现,特定的句型结构可以识别包含某种特定质量属性的用户评论。因此,文中针对表 1 的代码质量属性,综合考虑词与词性的特点,定义了识别不同质量属性的评价句型规则;然后,应用基于评价句型的代码质量属性识别规则对具有代码质量属性的用户评论进行识别。基于评价句型的质量属性识别规则及示例如表 5 所列。

鉴于只有部分代码质量属性可以根据用户评论规律定义出评价句型质量属性识别规则,且只有该规则对应的代码质量属性的用户评论能被识别,为了尽可能识别出更多的具有代码质量属性的用户评论,在基于评价句型识别具有质量属性的用户评论规则的基础上,针对代码的用户评论,根据第 3 节中建立的代码质量特征词库,应用评价对象质量属性识别规则识别具有质量属性的用户评论。本文在实验中发现,基于评价对象识别具有质量属性的用户评论规则可以识别出更多的用户评论。但是,如果只根据有限词汇挖掘用户评论,将导致应用语义空间挖掘出的用户意图信息有限^[14]。而且,随着数据量的快速增长,通过人工分析数据搜集的特征词十分有限,而通过特征词的相似词或同义词扩展特征词库来挖掘用户评论,会使针对用户评论的挖掘方法局限于特征词,忽略了句型、文本整体语义结构的重要性^[16]。

表 5 基于评价句型的质量属性识别规则及示例

评价句型	约束	用户评论示例	质量属性
v+[d/u/a]+...	v={看}	看/v 不/d 懂/v	可读性
ry+...+v	v={配置,部署}	tomcatjvm/x 参数/n 咋/ryv 设置/v	逻辑性
...v+v+ry	无	代码/n 是/v 干/v 啥/ry	功能性
d+vf+...	d={不}	为/n 啥/ry 运行/vi 结果/n 显示/v 不/d 出来/vf	运行性能
ryv+...+v	v={解决}	出现/v 异常/a 怎/ryv 么/nrl 解决/v	可靠性
d+v+...	d={不} v={支持}	不/d 支持/v locations/n	可移植性
ryv+nrl+v+...	v={用}	请问/v 接口/n 怎/ryv 么/nrl 用/v 呢/y	易用性

综上所述,本文首先应用评价句型规则识别具有代码质量属性的用户评论,再应用评价对象特征词库识别具有代码质量属性的用户评论。通过这两种识别规则,可以识别出大部分具有代码质量属性的用户评论。基于评价对象和评价句型识别具有代码质量属性的用户评论的流程如图 1 所示。

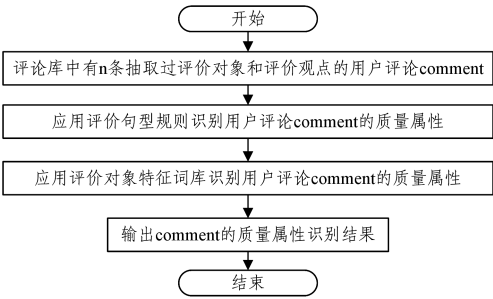


图 1 识别具有代码质量属性的用户评论的流程
Fig. 1 Recognition flowchart of user's comments with code quality attributes

本文的核心是对具有代码质量属性的用户评论进行识别、提取,并对代码质量进行分析;通过识别具有代码质量属性的用户评论,过滤无关的用户评论,以提高代码质量的分析效率和相关结果的有效价值。

5 代码质量分析

数据分析目的是将隐藏在一大批杂乱无章的数据中的信息集中并将其提炼出来,从而找出研究对象的内在规律。在实际应用中,数据分析可帮助人们作出判断,以便采取适当行动。第 4 节只筛选了与代码质量属性相关的用户评论,未进行分析处理,数据中隐藏的有价值信息没有被挖掘出来。因此,为了深入研究代码静态和动态质量,需要对具有代码质量属性的用户评论做进一步的分析。

为了能够更准确地获取代码使用者对代码质量的关注点,本文针对同一代码段的多条用户评论进行统计,获取用户最关注的每段代码的质量属性。据此,从具有代码质量属性的用户评论中抽取用户最关注的质量属性表现,这些表现有助于代码使用者判断代码质量是否满足要求,帮助代码使用者进一步了解代码静态和动态质量的具体内容。

但是,由于针对代码的用户评论具有稀疏性,有噪声和口语术语,导致运用机器学习算法进行用户评论分析时会产生特征稀疏、粒度难以统一的问题,使其抽取用户评论信息的效果较差^[17]。而评价观点是用户评论中评价内容的情感体现,评价观点前后词性为名词 n、名词性语素 ng、动词 v、动名词

vn、不及物动词 vi 的词通常为其关联词,关联词构成了评价观点的评价主体。关联词与评价观点按照词语在评论句子中的顺序重新组合为评价的具体内容,即代码质量属性的具体表现。

通过分析大量用户评论发现,评价观点前后关联词的词性为名词 n 或名词性语素 ng 时,抽取的质量属性表现一般为

静态质量属性表现;而评价观点前后关联词的词性为动词 v、动名词 vn 或不及物动词 vi 时,抽取的质量属性表现一般为动态质量属性表现。因此,为了抽取质量属性表现,本文结合评价观点抽取规则,在评价观点抽取的基础上,提出了一个基于评价观点的代码质量属性表现抽取方法,其流程如图 2 所示。

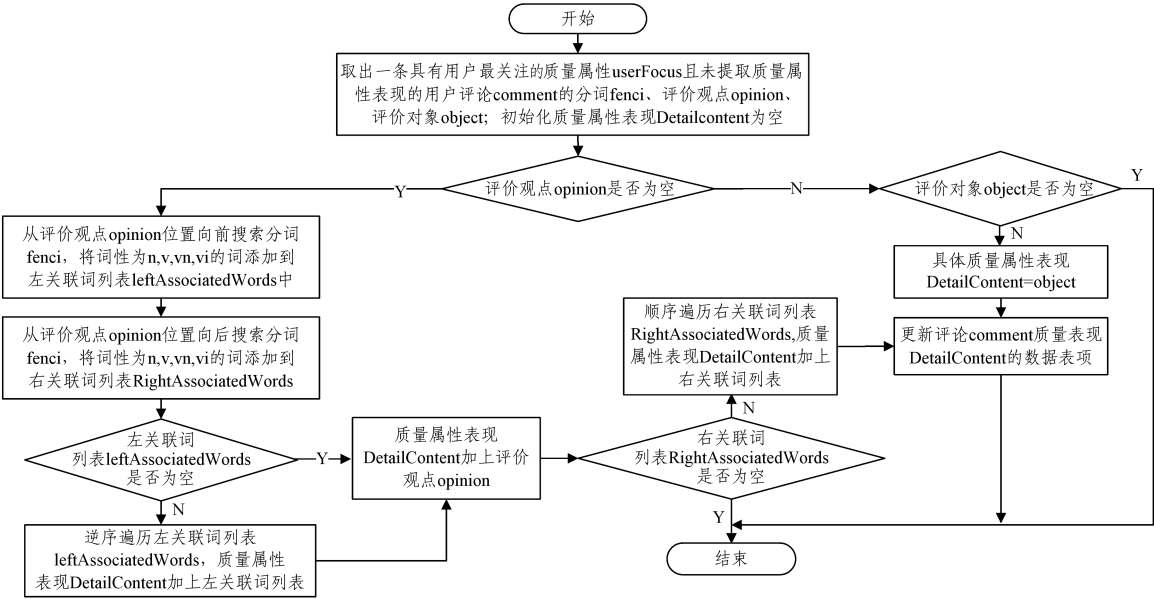


图 2 抽取代码质量属性表现方法的流程

Fig. 2 Flowchart of extraction for code quality attribute representation

应用质量属性表现抽取规则时,首先搜索评价观点前后词性为 n, ng, v, vn, vi 的关联词,然后按照关联词和评价观点在评论句子中原本的顺序进行重新组合,得到用户评论的代码质量属性表现。

由于抽取的质量属性表现复杂且散乱,无法直观地为代码提供者反馈用户对代码质量的关注点,且无法为代码使用者提供选择代码的参考点,文献[13]在挖掘体现反馈的 APP 的用户评论时,应用评论种子挖掘具有相似度的用户评论,其与评论种子体现相同反馈类型。

本文应用相似度算法^[18],针对同一段代码的多条用户评论抽取出质量属性表现并计算其相似度,对具有相关性的质量属性表现进行合并提取类别,结合情感倾向分析,获取代码静态质量和动态质量的相关结果。代码质量的用户情感倾向分析公式如式(1)所示:

$$S(QA) = \frac{(p_1 + p_2 + \dots + p_n)}{n} \tag{1}$$

其中, S(QA) 表示用户对代码质量属性 QA 的情感极值, p_n 表示用户对代码质量属性 QA 的表现情感极值, n 表示代码质量属性 QA 表现的数量。

分析用户评论中的质量属性表现及情感倾向,不仅有助于代码使用者判断代码质量是否满足要求,更能使其深入研究用户对代码静态和动态质量的关注内容,帮助代码提供者提高代码质量。

6 实验分析

本文从 CSDN¹⁾ 和开源中国推出的代码托管平台 Gitee²⁾ 爬取了与代码相关的用户评论,包括代码段名称、用户 Id、用户评论内容与评论时间(精确到秒)等。在实验之前,根据用户 Id 和评论时间对爬取的数据进行去重,最终以 29 019 个用户针对 5 727 个代码段进行评论产生的 33 169 条用户评论作为实验对象,经过人工标注后,有 25 750 条用户评论含有代码质量属性。

为了分析本文方法的有效性,应用查准率(Precision)和召回率(Recall)作为本文方法识别具有代码质量属性的用户评论的有效性的评价指标^[19],并用识别率(Recognition)作为代码质量模型选取的特征词的有效性的评价指标。查准率表示识别出的用户评论中,质量属性识别正确的用户评论所占的比例;召回率表示人工标注的质量属性结果中,被正确识别的用户评论所占比例;识别率表示人工标注的质量属性结果中,特征词识别出的具有代码质量的用户评论所占比例。查准率、召回率和识别率的计算公式如式(2)一式(4)所示:

$$Recall = AA / ST \tag{2}$$

$$Precision = AA / AT \tag{3}$$

$$Recognition = AC / ST \tag{4}$$

其中, AA 表示应用本文方法正确识别出的具有代码质量属性的用户评论数目; AT 表示所有识别出的具有代码质量属

1) <https://www.csdn.net>

2) <https://gitee.com>

性的用户评论总数目;AC 表示基于特征词规则识别出的用户评论数目;ST 表示人工标注的具有代码质量属性的用户评论数目。

6.1 识别具有代码质量的用户评论

针对查准率和召回率两个评价指标,本文对基于评价对象与评价句型识别质量属性的方法进行实验,并将其与基于TextCNN^[20]深度学习模型识别具有代码质量属性的用户评论的实验结果进行对比。

6.1.1 基于评价对象与评价句型识别用户评论

本节应用基于评价对象与评价句型识别质量属性的方法,从 33 169 条用户评论中识别出 23 811 条具有代码质量属性的用户评论。其中,基于评价对象识别出 18 031 条用户评论;基于评价句型识别出具体 5 780 条用户评论,识别出的具有代码质量的用户评论结果如图 3 所示。

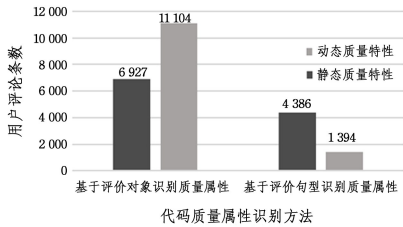


图 3 具有代码质量的用户评论识别结果

从图 3 可以看出,基于评价对象识别质量属性的识别效果优于基于评价句型的识别效果,与针对代码的用户评论具有强领域性的特点相符。此外,实验结果表明,在针对代码的用户评论中,用户更关注代码的动态质量。

通过与人工标注出的具有代码质量属性的 25 750 条数据对比,本文方法识别具有代码质量属性的用户评论的查准率为 77.4%,召回率为 71.5%。此外,在基于评价对象与评价句型识别具有代码质量属性的用户评论的两种规则中,基于特征词识别出具有代码质量属性的用户评论 18 221 条,识别率为 70.3%。综上,基于评价对象与评价句型的用户评论识别方法能够识别出具有代码质量的用户评论,且基于代码质量模型的特征词在识别具有代码质量的用户评论时是有效的。

6.1.2 基于 TextCNN 识别用户评论

为了进一步验证本文方法的有效性,针对人工标注出 25 750 条数据,采用基于 TextCNN 深度神经网络实现的文本分类模型进行对比实验,数据集划分如表 6 所列。

表 6 数据集划分
Table 6 Dataset partitioning

训练集	20 000
测试集	2 875
验证集	2 875

应用基于 TextCNN 的文本分类模型识别具有代码质量的用户评论的查准率为 66.8%,召回率为 60.3%。与基于 TextCNN 的文本分类模型相比,本文方法在识别具有代码质量的用户评论时效果更加显著。

6.2 代码质量分析

基于 6.1 节中识别出的具有代码质量的 23 811 条用户评

论,应用第 5 节论述的方法进行代码质量分析,相关结果如表 7 所列。

表 7 代码质量分析结果
Table 7 Analysis results of code quality

类型	质量属性	本文方法识别的评论条数	人工标注的评论条数	本文方法提取表现类别数量	人工标注表现类别数量	表现类别名称	情感倾向
静态质量	可读性	3430	4418	3	3	注释	0.41
						易懂	1.85
						代码	0.35
	编程规范性	1522	1239	3	3	编码格式	-0.02
						缩进	0.88
						utf8	0.25
	可重用性	4284	5683	2	2	源码	0.09
						数据库	0.21
						数据	0.02
	逻辑性	2077	1756	2	3	配置	-0.08
功能						0.33	
动态质量	功能性	1605	1975	6	5	登录失败	-0.15
						获取请求	0.04
						访问	-0.02
						页面	-0.49
						实现	-0.33
	可测试性	2107	1591	3	3	编译失败	-0.93
						问题	0.56
						测试	-0.12
	可维护性	1343	1239	2	3	接口	0.56
						更新	1.31
运行性能	1793	2273	3	3	运行	-0.09	
					效果	-0.19	
					代码运行	-0.03	
					bug	-0.51	
					解决方法	-0.05	
可靠性	2631	2483	5	5	问题	-0.23	
					报错	-0.53	
					版本	-0.09	
可移植性	1529	1580	3	3	环境	0.01	
					支持	-1.29	
					方便	0.56	
易用性	1490	1514	2	2	如何使用	-0.13	

表 7 中,“情感倾向”列数值为正表示用户的情感倾向为积极,数值为负表示用户的情感倾向为消极。从表 7 中本文方法和人工标注的评论条数结果可以看出,针对代码的用户评论中,用户最关注的静态代码质量是代码的可读性和可重用性,最关注的动态质量为代码运行性能和可靠性,本文方法与人工标注的结果一致。此外,根据情感倾向的用户情感值可以看出,用户对代码的动态质量的倾向多为负值,表明在针对代码质量的用户评论中,代码的动态质量一般不满足代码使用者的要求。从表 7 中本文方法提取表现类别数量和人工标注表现类别数量结果可以发现,在逻辑性和可维护性中,本文方法提取表现类别数量相对较少。因为代码的逻辑性和可维护性无法直接度量,导致在识别具有可维护性和逻辑性的用户评论时,部分用户评论无法被识别,从而在提取质量属性表现类别时不存在该类别的用户评论。此外,在功能性中,本文方法提取表现类别数量多于人工标注的表现类别数量。这是由于用户评论中代码功能性描述复杂多样,导致在功能性质量属性表现中,相似的质量属性表现数量较少,而不具有相似度的质量属性表现是单独划分为一类的,从而导致本文方法提取的表现类别数量多于人工标注的表现类别数量。

综上,本文方法提取质量属性表现类别的查准率为 94.1%,召回率为 91.4%,表明本文方法能够合并具有相似质量属性表现的用户评论并提取表现类别。

结束语 针对难以执行代码测试的情况,为了分析代码的静态质量和动态质量,本文首先对代码质量进行定义,提出了一个包含静态特性和动态特性的代码质量模型;然后,依据代码质量模型中的代码质量属性,识别具有代码质量的用户评论;最后,对抽取出的用户最关注的代码质量属性表现进行合并和提取类别,并结合情感分析得出代码质量分析结果。

实验表明,本文方法是有效的,但是由于建立的代码质量特征词库和评价句型规则不够完善,应用基于评价对象和评价句型的质量属性识别规则来识别表达复杂的用户评论质量属性时效果较差。下一步,将针对多分句表达复杂的用户评论开展相关研究。

参 考 文 献

- [1] PAGANO D, BRÜGGE B. User involvement in software evolution practice: a case study [C] // International Conference on Software Engineering. San Francisco: IEEE Press, 2013: 953-962.
- [2] LU Z, YANG D, LI J. A software evaluation system based on reviews mining[J]. Computer Applications and Software, 2014, 31(7): 1-4.
- [3] RADHIKA D, VENKATASUBRAMANYAM R D, SOWMYA G R. Why is dynamic analysis not used as extensively as static analysis: an industrial study [C] // Proceedings of the 1st International Workshop on Software Engineering Research and Industrial Practices, 2014: 24-33.
- [4] HUANG P J, YANG M Q. Research and Application of Static Metrics for Code Quality[J]. Computer Engineering and Applications, 2011, 47(23): 61-63.
- [5] LIU D R, LIU D S, ZHANG L P, et al. Prediction of Code Clone Quality Based on Bayesian Network [J]. Computer Science, 2017, 44(4): 165-168.
- [6] XU N C, YUAN J, GAO X L. Code-based IT Community Answer Quality Evaluation Model[J]. Journal of Chinese Computer Systems, 2019, 40(1): 158-163.
- [7] POCHE E, JHA N, WILLIAMS G, et al. Analyzing User Comments on YouTube Coding Tutorial Videos [C] // IEEE/ACM International Conference on Program Comprehension. IEEE, 2017.
- [8] ZHANY Y H, ZHU X F, XU C Y, et al. Hybrid Recommendation Approach Based on Deep Sentiment Analysis of User Reviews and Multi-View Collaborative Fusion[J]. Chinese Journal of Computers, 2019, 42(6): 1317-1333.
- [9] FENG Y, CHEN Y G, QIONG B H. Social and Comment Text CNN Model Based Automobile Recommendation[J]. Acta Automatica Sinica, 2019, 45(3): 76-87.
- [10] ZHANG Y, WANG C, GUO W Y, et al. Multi-Source Emotion Tagging for Online News Comments Using Bi-Directional Hierarchical Semantic Representation Model[J]. Journal of Computer Research and Development, 2018, 55(5): 43-54.
- [11] CHEN Q, ZHANG L, JIANG J, et al. Review Analysis Method Based on Support Vector Machine and Latent Dirichlet Allocation[J]. Journal of Software, 2019, 30(5): 349-362.
- [12] TOLL D, ASSEMA J V, DURAN R, et al. I know it when I see it: Perceptions of Code Quality [C] // ACM Conference on Innovation & Technology in Computer Science Education. ACM, 2017.
- [13] HU T Y, JIANG Y. Mining of User's Comments Reflecting Usage Feedback for APP Software [J]. Journal of Software, 2019, 30(10): 3168-3185.
- [14] ZHENG R J. Computer Software Testing Technology [M]. Beijing: Tsinghua University Press, 1992: 31-35.
- [15] ATOUM I, BONG C H, KULATHURAMAIYER N. Towards resolving software quality-in-use measurement challenges [J]. Computer Science, 2015, 5(11): 877-885.
- [16] LEOPAIROTE W, SURARERKS A, PROMPOON N. Software quality in use characteristic mining from customer reviews [C] // Digital Information and Communication Technology and Applications (DICTAP). 2012: 434-439.
- [17] ZHANG H B, ZHONG H, HU X L. User Reviews Clustering Method Based on Topic Analysis [J]. Computer Science, 2019, 46(8): 50-55.
- [18] LEVENSHTAIN V. Binary codes capable of correcting deletions, insertions, and reversals [J]. Soviet Physics Doklady, 1966, 10(8): 707-710.
- [19] DENG H C, WANG M F. Theoretical Study of the Relationship between Recall and Precision Ratio [J]. Journal of The China Society for Scientific and Technical Information, 2000, 19(4): 359-362.
- [20] KIM Y. Convolutional neural networks for sentence classification [C] // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar, 2014: 1746-1751.



XU Hai-yan, born in 1996, postgraduate, is member of China Computer Federation. Her main research interests include software engineering, software quality assurance and testing.



JIANG Ying, born in 1974, Ph.D, professor, Ph.D supervisor, is senior member of China Computer Federation. Her main research interests include software quality assurance and testing, cloud computing, big data analysis and intelligent software engineering.