软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(12):3648−3664 [doi: 10.13328/j.cnki.jos.005329]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

众测中的工作者选择方法研究*

崔 强[1,3], 王俊杰[1], 谢 淼[1,3], 王 青[1,2,3]

[1](中国科学院 软件研究所 互联网软件技术实验室,北京 100190)

[2](计算机科学国家重点实验室(中国科学院 软件研究所),北京 100190)

[3](中国科学院大学,北京 100049)

通讯作者: 王青, E-mail: wq@itechs.iscas.ac.cn

摘 要: 众测是一种新兴的软件测试方法,它依靠网络上的工作者帮助完成测试任务.对于某个测试任务来说,谁来执行对于发现缺陷以及覆盖测试需求关键点是至关重要的.然而众测平台上一般有大量的候选工作者,他们拥有不同的测试经验,也常常提交重复的测试报告.由于众测工作者随机地参与测试任务,同时满足较高缺陷检测率和较高测试需求关键点覆盖度是很困难的.因此,该文关注如何为新的测试任务选择一组合适的众测工作者,从而提高缺陷检测率和需求关键点覆盖度.首先设计了 3 个实验,试图发现选择什么样的众测工作者能够提升缺陷检测率和需求关键点覆盖度.通过实验验证,发现众测工作者的主动性、相关性和多样性从不同的角度影响测试质量,并且给出了它们的度量方法.然后,提出一种同时考虑这3个方面工作的选择方法.基于众测平台之一——百度众测上46个真实的测试任务对该方法进行了验证,结果显示,该方法能够显著提高缺陷检测率和测试需求关键点覆盖度.

关键词: 众测;缺陷检测;需求覆盖;人员选择

中图法分类号: TP311

中文引用格式: 崔强,王俊杰,谢淼,王青.众测中的工作者选择方法研究.软件学报,2018,29(12):3648−3664. http://www.jos.org.cn/1000-9825/5329.htm

英文引用格式: Cui Q, Wang JJ, Xie M, Wang Q. Towards crowd worker selection for crowdsourced testing task. Ruan Jian Xue Bao/Journal of Software, 2018,29(12):3648−3664 (in Chinese). http://www.jos.org.cn/1000-9825/5329.htm

Towards Crowd Worker Selection for Crowdsourced Testing Task

CUI Qiang[1,3], WANG Jun-Jie[1], XIE Miao[1,3], WANG Qing[1,2,3]

[1](Laboratory for Internet Software Technologies, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

[2](State Key Laboratory of Computer Science (Institute of Software, The Chinese Academy of Sciences), Beijing 100190, China)

[3](University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: Crowdsourced testing is an emerging trend in software testing, which relies on crowd workers to accomplish test tasks. Thus, who performs a test task is extremely important for detecting bugs and covering key points of test requirements in crowdsourced testing. There are a lot of candidate crowd workers who may have different testing experience but can also produce duplicate test reports for the same task due to the lack of cooperation. As crowd workers can freely participate in a test task, high quality of testing in terms of bug detection and coverage of key points of test requirements is not guaranteed. Thus, to improve bug detection and coverage of key points of test requirements, selecting an appropriate subset of workers to perform a test task is becoming an important problem. In this paper, three motivating studies are first conducted to investigate important aspects of workers in detecting bugs and covering key points of test requirements. Accordingly, the studies identify three aspects: initiative, relevance and diversity are identified, and produce a novel

* 基金项目: 国家自然科学基金(61602450, 61432001)
Foundation item: National Natural Science Foundation of China (61602450, 61432001)
收稿时间: 2016-12-20; 修改时间: 2017-03-10; 采用时间: 2017-06-19

approach for selecting workers considering all these three aspects. This new approach is evaluated based on 46 real test tasks from Baidu CrowdTest, and the experimental results show the effectiveness of the approach.

Key words: crowdsourced testing; bug detection; requirement coverage; worker selection

作为一种新兴的软件测试方法,众测正越来越受到来自工业界和学术界的关注[1−3].很多成功众测平台(例如 uTest、Pay4Bugs、百度众测等)的兴起,使得测试活动变得更高效.在众测中,众测任务被发布到众测平台上,众测工作者(crowd worker)可以选择感兴趣的任务执行,并提交测试报告(test report)来描述被测软件系统的行为和发现的缺陷.通过这种形式,众测工作者帮助了公司的开发和测试工程师发现软件系统的缺陷.在众测中,对缺陷进行检测以及对测试需求关键点进行覆盖都是很重要的.

与传统测试活动相比,众测带来的显著改变是将众测任务交给众测工作者完成,他们可以位于不同的地域,对于软件测试也有着不同的经验和技能.虽然在众测中有很多的候选工作者,但由于成本的限制,不可能允许所有人都来执行测试任务(test task).因此,谁来执行某个测试任务,对于缺陷发现和测试需求关键点覆盖是至关重要的.因为众测工作者的经验不同,对于某个测试任务,有些工作者能够检测到缺陷,而有些工作者不能检测到缺陷.进一步地,有些工作者可能发现的是重复的缺陷,这会导致不必要的成本开销.然而在实际的众测中,由于众测工作者随机地选择测试任务,这些任务较高的缺陷检测率和较高的需求关键点覆盖度并不能得到保证.因此,本研究聚焦如何为新的测试任务选择一组众测工作者,从而提高缺陷检测率和测试需求关键点覆盖度,这对于减少测试成本、提高测试效率是至关重要的.

我们首先设计了 3 个实验,试图研究选择什么样的众测工作者可以提升缺陷检测率和测试需求关键点覆盖度.根据实验结果,我们发现:

(1) 提交测试报告多的工作者,也就是有着高主动性的工作者,更可能发现缺陷;

(2) 与测试任务有着更多相似经验的工作者,也就是经验更相关的工作者,更可能发现缺陷;

(3) 一组有着不同经验的工作者,也就是更多样化经验的工作者,更能够发现缺陷.

基于上述发现,我们从 3 个方面来刻画众测工作者,分别是主动性、相关性和多样性.主动性通过工作者提交的测试报告数目度量;相关性通过测试需求和工作者经验之间的相似性度量;多样性通过一组工作者之间的经验的差异性来度量.

我们提出一种为众测任务选择工作者的方法,能够同时考虑主动性、相关性和多样性这 3 个方面.首先基于工作者历史提交的测试报告来建模工作者,基于测试需求的文本描述建模测试任务.然后,给出以上 3 个方面的度量方法.最后,给出如何为特定的众测任务选择一组工作者的选择算法.

基于百度众测对方法进行验证的结果显示,相比其他 3 个基线方法,本文方法在提高测试需求关键点的覆盖度和缺陷检测率方面有着显著的作用.我们还用实验说明了主动性、相关性和多样性这 3 个方面在人员选择上是互相补充的,并且还模拟了实际应用场景,对本文方法在实际应用场景中的性能进行验证.

本文第 1 节给出众测的背景、一些基本概念以及数据集.第 2 节介绍 3 个启发式实验及结果.第 3 节给出工作者选择的方法.第 4 节验证方法的有效性.第 5 节对方法和结果进行讨论,并给出对结果有效性的威胁.第 6 节介绍相关工作.第 7 节总结全文.

1 众 测

1.1 众测基本流程

如图 1 所示,测试任务由任务发布者发布到众测平台上,众测工作者从众测平台上选择测试任务,并且下载到自己的设备上完成测试,测试完成后,众测工作者提交测试报告,测试报告经由测试平台最终反馈给任务发布者.最后,测试发布者根据测试报告反馈的缺陷改进软件,并支付众测工作者相应的报酬.值得一提的是:不是每个测试报告都涉及缺陷,并且有些测试报告涉及的是同一缺陷(这些测试报告被称为重复报告).众测与传统的开源社区不同,众测工作者可以从参加并完成的任务中获得一定的报酬,这一方面吸引了更多的工作者,另一方

面也让众测必须考虑到成本因素.

Fig.1 Procedure of crowdsourced testing

图 1 众测的基本流程

众测工作者选择测试任务有两种模式:一种是自取(pull)模式,工作者自己随机寻找测试任务进行测试;另一种是推荐(push)模式,众测平台向一组工作者推荐某个测试任务,他们根据自身情况决定是否进行测试.很多众测平台都是基于自取模式,但该模式存在一些弊端.众测工作者需要花费大量的时间寻找任务,并且可能执行他们不擅长的测试任务,导致产生一些低质量的测试结果.因此,推荐模式逐渐兴起,人们也越来越意识到该模式的价值.例如:百度众测会为工作者提供测试任务推送功能,通知最近上线和适合工作者参与的众测任务,该功能给工作者推送全部他们等级可见的众测任务的上线消息,大范围发送,并没有定向推送和选择;Testin 为每个众测任务提供专家分配的服务.一方面,这些推荐模式的服务减少了工作者搜索任务的时间,另一方面,可以针对众测任务的特点分配合适的工作者.该功能主要基于测试专家的详细背景资料(包括能力、专业等),绝大部分众测平台上并没有如此详细的背景信息.然而在绝大多数众测平台上,关于将某测试任务推荐给哪些测试人员,从而提高缺陷检测的效率,目前并没有很好的解决方案.本文以此为出发点,提出了相应的解决方法.

1.2 基本概念

本文涉及众测中的 3 个基本概念,分别是测试任务、测试报告、众测工作者.

- 测试任务:指发布在众测网站上的软件测试众包任务,包括任务发布者的信息以及由测试任务发布者提供的自然语言描述的测试需求.测试任务表示为一个由技术术语组成的向量,这些技术术语是从测试需求的自然语言描述中抽取得到的:$tk=\{t_1,t_2,\ldots,t_i,\ldots\},t_i\in T$($T$ 是技术术语词表);
- 测试报告:指工作者完成测试后,反馈给任务发布者的报告.它包括测试输入、操作步骤、软件行为的描述以及工作者给出的是否为缺陷的标记.与测试任务类似,测试报告也表示为一个由技术术语组成的向量,这些技术术语是从测试报告的所有文本描述中抽取得到的:

$$tr=\{t_1,t_2,t_3,\ldots,t_i,\ldots\},t_i\in T;$$

- 众测工作者:指众测平台上注册的愿意执行测试任务的用户,每个众测工作者都伴随着他们的身份信息以及历史提交的测试报告.本文用工作者历史提交的测试报告来刻画他们的经验.与测试任务类似,众测工作者也表示为一个由技术术语组成的向量,这些技术术语是从该工作者历史提交的所有测试报告中抽取得到的:$w=\{t_1,t_2,t_3,\ldots,t_i,\ldots\},t_i\in T$.

本文还涉及两个与众测相关的概念——缺陷检测率、测试需求关键点覆盖度.

- 缺陷检测率(bug detection rate)定义为一组被选中的众测工作者检测到的缺陷占所有缺陷的比例;
- 测试需求关键点的工作者覆盖度(worker coverage of key points of test requirements)定义为一组被选中的众测工作者历史提交的测试报告覆盖的技术术语占测试需求中所有技术术语的比例.

这些基本概念会在后文用到,尤其是第 2 节和第 3 节.我们会在第 3.1 节详细介绍是如何抽取技术术语的.

1.3 百度众测数据集

本文用到的实验数据集是从百度众测平台收集到的,该平台成立于 2011 年,是国内最大的工业众测平台之一,包含上万名众测工作者、上千款机型,支持移动测试、桌面测试、网页测试等多种类型的测试.我们收集到 2015 年 11 月 1 日~11 月 30 日期间关闭的所有测试任务,共有 46 个测试任务,涵盖 8 个类别(例如音乐、工具、游戏等).以上众测任务关闭的时候,绝大多数测试任务都被测试得比较充分,即,绝大部分缺陷都已经被检出.

每个提交的测试报告已经被管理人员审核,并且标记上“缺陷”、“非缺陷”的标签.此外,对于多个报告描述同一缺陷的情况,管理人员也都标记上“重复”的标签.我们基于这些标签来验证本文方法.另外,既然重复缺陷对于软件质量的改进不能带来帮助,当统计一组人员检测到的缺陷数目时,我们将重复的缺陷记为 1 个.

本文的实验数据集包括 46 个测试任务,844 个众测工作者,3 984 个测试报告.如图 2 所示,众测工作者提交的测试报告呈帕累托分布,大多数工作者只提交 1 个测试报告,只有少数工作者提交大量的测试报告.平均来看,每个测试任务会有 37 个提交的测试报告,并且每个测试任务平均检测到 15 个缺陷.

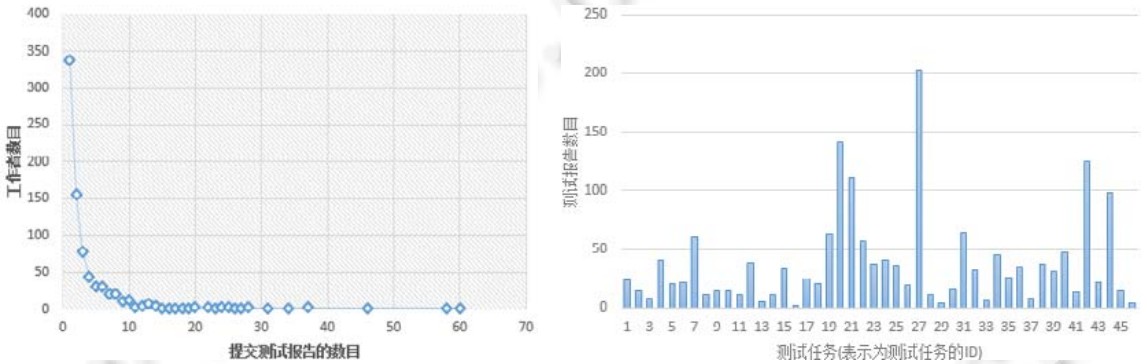


Fig.2 Statistics of dataset

图 2 数据集统计

2 动机实验

本节研究众测环境下,如何选择工作者能够提升缺陷检测率和测试关键点需求覆盖.其中,提高测试需求覆盖比较简单直接,即选择有着相关测试经验的工作者,并且保证所选择的工作者测试经验的多样化即可.但是如何选择工作者能够提升缺陷检测率需要进一步研究,因此我们设计 3 个实验主要研究如何选择工作者能够提升缺陷检测率.

2.1 提交测试报告多的人会发现更多缺陷吗?

- 动机

通过分析数据集,我们发现一些工作者提交了很多的测试报告,而有些工作者只提交了很少的报告.直观上看,提交很多报告表明工作者愿意投入时间和精力进行测试,从而更可能发现缺陷.因此,我们研究工作者提交的报告数目和发现的缺陷数目之间的关系.

- 方法

首先,我们按照众测任务完成时间把 46 个测试任务按照先后顺序分配到 5 个时间桶中,这里,每个时间桶包含基本相等数量的众测任务(时间桶 0~4 分别包含 10,9,9,9,9 个测试任务),由此,5 个桶之间产生 4 个时间点,我们分别统计在每个时间点之前工作者提交的测试报告数目和他在该时间之后发现缺陷数目.然后,在每个时间点上,我们通过皮尔逊相关系数分析工作者之前提交的测试报告数目与他在其后发现缺陷数目的相关关系.图 3 给出每个时间点的皮尔逊相关系数.

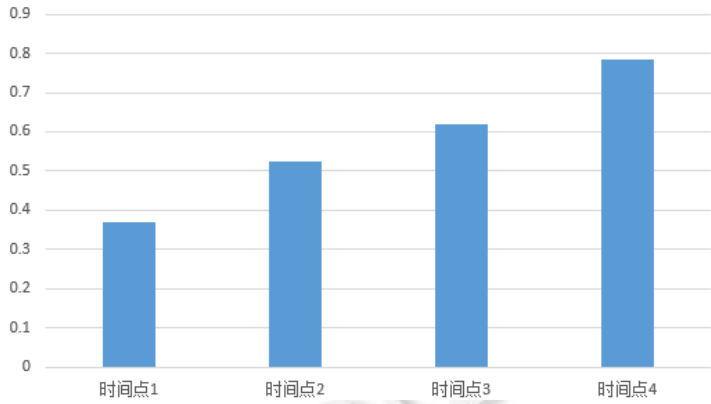


Fig.3 Pearson correlation coefficient between number of submitted reports and number of detected bugs in differernt time points

图 3 不同时间点上工作者提交的测试报告数和发现缺陷数的皮尔逊相关系数

- 结果

从图 3 可以看出:随着历史数据的累积,工作者提交的测试报告的数目和发现的缺陷数目的相关性逐渐上升.在全部 4 个时间点上,皮尔逊相关系数都显示,工作者提交测试报告数和发现缺陷数目正相关,其中,最小的皮尔逊相关系数为 0.37(p-$value$<0.05),最大为 0.785(p-$value$<0.05).以上结果表明,众测工作者提交的测试报告数目与发现缺陷数目存在极强相关性.由此,提交越多测试报告的工作者,会发现越多的缺陷.这个结论听起来比较普通,然而它可以指导我们进行人员选择,从而在众测环境下发现更多的缺陷.提交越多测试报告的工作者,越可能在众测任务中发现缺陷.

2.2 与测试需求有着相似经验的人会发现更多缺陷吗?

- 动机

我们通过分析每个众测工作者提交的测试报告,发现很多工作者检测到的缺陷均为软件的某些相似功能.例如:某个工作者发现的很多缺陷都是与定位功能相关的,另一工作者发现的很多缺陷都是与展示功能相关的.该发现表明:众测工作者通过执行测试任务,积累某些特定方面的经验,进而变得擅长某些任务.另一方面,由于测试任务大多为专业化的任务,最好让工作者执行他们擅长的任务,这样可以提高缺陷检测率.因此,我们试图研究是否与测试需求有着更相似经验的工作者能够发现更多的缺陷.

- 方法

对于每个测试任务,我们首先计算对应的测试需求和参与执行该任务的每个工作者的经验之间的文本相似性.同时,我们将该任务涉及的工作者分为两组:一组是缺陷组,包含在该测试任务中检测到缺陷的工作者;一组为非缺陷组,包含在该测试任务中没有检测到缺陷的工作者.然后在每组内,计算工作者和测试需求的相似性的平均值.对于每个测试任务,基于公式(1),得到两个相似平均值之间的相对差异($rd(a,b)$):

$$rd(a,b)=(a-b)/\min(a,b) \tag{1}$$

其中,a 为缺陷组的所有工作者的经验和测试需求之间的相似性的平均值,b 为非缺陷组的所有工作者的经验和测试需求之间的相似性的均值.

- 结果

图 4 给出了所有项目的相对差异值.在横轴以上的柱状图表示缺陷组的工作者经验和测试需求之间的平均相似性比非缺陷组的更高,也就是说,对于有着与测试需求更相关的测试经验的工作者更容易发现软件缺陷.在 71%(33/46)的测试任务中,柱状图都在横轴以上,最大的平均相似性差异可以达到 2 倍.这意味着,工作者经验和测试需求之间的相似性关系有利于在测试任务中检测出缺陷.也有一些柱状图是在横轴以下的,但是他们数

目较少,并且数值也相对较小.与测试需求有着越相似经验的众测工作者,越可能在众测任务中发现缺陷.

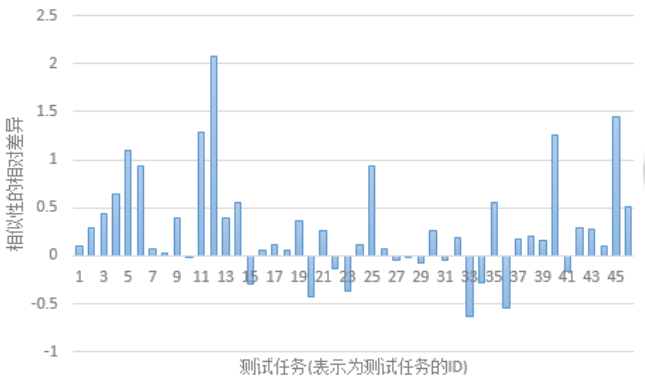


Fig.4 Relative difference of similarity between test requirements and worker experience in BUG group and NON-BUG group

图 4 测试需求和工作者的经验相似性在发现缺陷组合未发现缺陷组的相对差异

2.3 不同经验的人会发现不同的缺陷吗?

- 动机

在每个测试任务中,都会存在一些重复的测试报告.这些重复报告描述的是同一个缺陷,因此,对于测试和质量改进没有作用.我们试图分析是否不同经验的众测工作者会发现不同的缺陷.

- 方法

首先,将每两个缺陷归为一个缺陷对,并且计算每个缺陷对的测试报告的文本距离.同时,我们用 CLUTO 聚类工具(http://glaros.dtc.umn.edu/gkhome/views/cluto),根据工作者的经验对其进行聚类,如第 1.2 节所述,我们将工作者表达为他的历史测试报告中技术术语所组成的向量,采用 Kmeans 算法对工作者进行聚类分析.对于每个任务,我们将缺陷对分为两组:一是相似组,包含由同一个类簇中的两个工作者发现的两个缺陷组成的缺陷对;二是非相似组,包含不同类簇中的两个工作者发现的缺陷对.然后对于每个任务,计算两组中所有缺陷对的测试报告的距离平均值,并根据公式(1),得到相对差异.这里,相似组的平均距离为 b,非相似组的平均距离为 a.

- 结果

图 5 给出了所有项目的相对差异值.横轴以上的柱状图表示由不同类簇中的工作者发现的两个缺陷报告的平均文本距离更大.我们发现:对于 72%(33/46)的测试任务,不相似组的缺陷平均距离比相似组的平均距离更大.此外,在 38%(18/46)的测试任务中,平均距离的相对差异都在 10%以上.有 21%的任务柱状图在横轴以下,但其相对差异值较小,很多几乎为 0.这表明,有着不同经验的工作者更可能发现不同的缺陷.也就是说,更多样的众测工作者能够帮助发现更多的缺陷.不同经验的众测工作者更可能发现不同的缺陷.

2.4 总 结

以上 3 个实验说明了什么样的众测工作者更可能在众测环境下发现缺陷.我们得到以下 3 个启示.

(1) 众测工作者提交的测试报告的数目反映了他的主动性.实验结果表明,提交测试报告多的工作者会发现更多的缺陷.这说明主动性能够用来区分工作者的缺陷检测能力;

(2) 有着与测试需求更相似经验的众测工作者更能够发现缺陷.这说明对于某个测试任务,存在一些相关的工作者,他们更能够在该任务中发现缺陷.因此,测试需求和工作者经验之间的相关性对于选择和测试任务匹配的工作者是至关重要的,也能够增加缺陷发现的概率;

(3) 不同经验的工作者更能够发现不同的缺陷.这说明多样化的工作者有助于减少重复缺陷报告的数目.因此,多样性对于选择一组工作者从而覆盖测试需求和提高缺陷检测效率是很关键的.

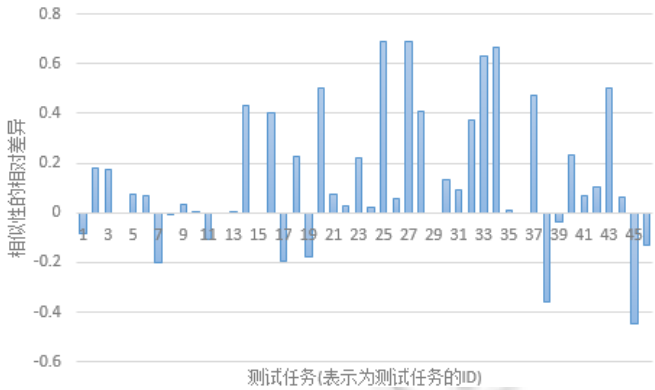


Fig.5 Relative difference in distance of two bugs which detected by workers in SIM group or NON-SIM group

图 5 缺陷描述的相似性在相似组和非相似组的相对差异

3 众测工作者选择方法

第 2 节的实验表明:众测工作者的 3 个方面对于发现缺陷是有用的,我们定义这 3 个方面为主动性、相关性和多样性.主动性表征了工作者参加某个测试任务的意愿;相关性表征了测试任务的需求和工作者经验之间的相似性;多样性表征了工作者经验方面的差异.因此,这 3 个方面需要同时考虑,以便优化缺陷检测率和测试需求覆盖度.

本文的目标是为新的测试任务选择一组合适的工作者,通过最大化主动性、相关性、多样性这 3 方面,从而提高缺陷检测率和测试需求覆盖度.

我们定义该问题为:给定一个测试任务 tk、一组候选工作者 $W=\{w_1,w_2,w_3,\ldots,w_n\}$、每个工作者的历史测试报告集合 TR 以及一个整型参数 $k(k\leqslant n)$,目标是从候选集合 W 中选择出一个包含 k 个工作者的子集 S,使得选择出来的集合 S 中,每个工作者具有较大的主动性、与测试任务有较大的相关性,整个集合的工作者具有较大的多样性.

图 6 中给出方法的框架.首先,我们建模测试任务和众测工作者;然后,基于历史数据度量主动性、相关性和多样性这 3 个方面;最后提出一个选择算法来为给定的测试任务选择一组工作者.

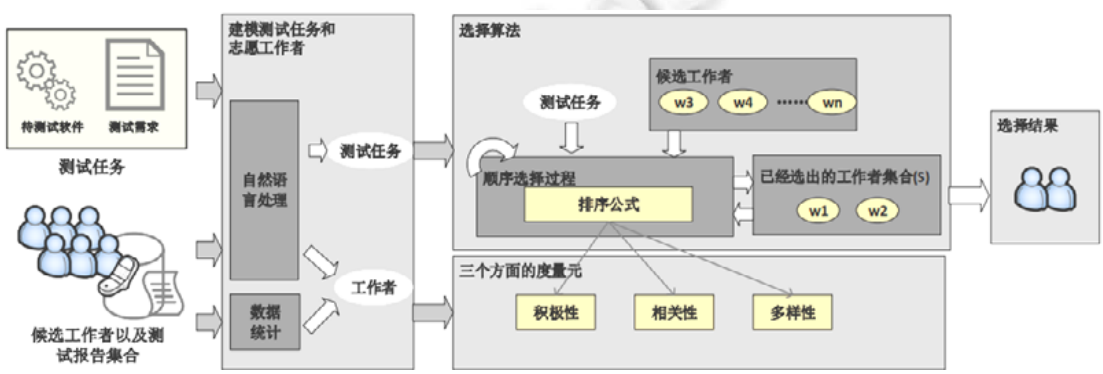


Fig.6 Overview of approach

图 6 方法框架

3.1 建模测试任务和众测工作者

我们用自然语言处理技术从测试需求和测试报告中抽取技术术语.为便于说明,每个测试需求或测试报告都被称为一个文档.

由于数据集的文档是中文形式,我们应用中国科学院计算技术研究所的分词工具 ICTCLAS (http://ictclas.nlpir.org)将每个文档切分为单词.为了减少噪音,我们基于中国知网提供的停用词表将停用词(例如的、了)去掉.然后,将剩余单词进行词性标注.已有研究表明,名词和动词对于文档的意义起着决定性作用[1].因此,我们只保留名词和动词.整个数据集得到的所有名词和动词构成了初始技术术语词表.

我们发现:有一些技术术语会出现在极大量的文档中,或只出现在极少量的文档中,这些技术术语在建模测试任务和工作者经验时作用较小.通过统计初始技术术语词表中每个技术术语出现在文档中的次数(通常被称为文档频率),过滤掉文档频率最高的 5%和最低的 5%的技术术语.因为频率最高的技术术语通常都是意思很宽泛的术语(如软件、返回等),对于表达测试需求和工作者经验作用很小且会引入一些噪音;频率最低的技术术语都只出现过 1~2 次(如能源、起飞),这些技术术语出现频率太低,对于计算相关性和多样性的作用很小,但是数量却很多.依照已有的众测相关工作[1]和 NLP 相关工作[26],我们过滤频率最高和最低的技术术语.剩余的技术术语构成了最终的技术术语词表,称为 T.

3.2 度量主动性、相关性、多样性

- 主动性($I(w_i)$)

我们用某众测工作者提交的测试报告的数目来度量其主动性.参考已有的度量工作者意愿和能力的工作[29],我们尝试了多种度量方式来度量主动性,包括参与任务数、发现缺陷数、提交报告数.结果发现,以上 3 种指标效果很相似,因为提交报告数可以比较细致地反映工作者参与测试任务的意愿和能力,我们最终选择了提交报告数来度量主动性:

$$I(w_i)=\#tr_i \tag{2}$$

- 相关性($R(tk|w_i)$)

我们用某众测工作者和测试需求之间的相似性程度来度量其与测试任务的相关性.我们用语言模型[5]来表示两者的相似性程度:

$$R(tk \mid w_i)=\prod_{t_j \in tk} P(t_j \mid w_i) \tag{3}$$

其中,t_j 是测试任务 tk 的测试需求描述中的技术术语.$P(t_j|w_i)$表示技术术语 t_j 出现工作者 w_i 的历史测试报告中的技术术语集合中的概率.详细来说, $P(t_j \mid w_i)=\frac{tf(t_j,w_i)}{\sum_t tf(t,w_i)}\cdot\frac{\sum_t df(t)}{df(t_i)}$.其中,$tf(t_j,w_i)$表示工作者 w_i 的历史测试报告的技术术语集合中术语 t_j 出现的次数,$df(t_j)$表示技术术语 t_j 总共在多少个工作者的历史测试报告的技术术语集合中出现.

- 多样性($D(w_i,S)$)

我们用众测工作者经验的差异性来度量一组工作者的多样性.考虑一个顺序的工作者选择过程,也就是每次从候选工作者集合中选择一个工作者.在此场景下,我们用某候选工作者 w_i 和已经选择出来的工作者集合 S 的差异程度来度量多样性.在工作者选择的每次迭代中,多样性指标 $D(w_i,S)$用于表征候选工作者 w_i 和已经选择出来的工作者集合 S 的差异程度.

详细来说,$D(w_i,S)$的度量是基于众测工作者涉及的技术术语.如果某些技术术语在已经选择出来的工作者集合的历史测试报告的技术术语集合中被提及过,那么这些术语的权重会降低.因此,与已经选择出的工作者集合相比,$D(w_i,S)$被度量为某工作者的经验涉及的新技术术语的程度.

我们首先定义 p_j 为技术术语 t_j 没有被已经选择出来的工作者集合 S 覆盖的程度:

$$p_j=\prod_{w_i \in S}(1-P(w_i \mid t_j)) \tag{4}$$

根据定义,p_j 值越大,表明技术术语 t_j 越没有被工作者集合 S 很好地覆盖,因此在选择工作者时,这些术语应

该有着更高的优先级.因此,$D(w_i,S)$被定义为:

$$D(w_i,S)=\sum_{t_j\in T}P(w_i\mid t_j)\times p_j \tag{5}$$

根据该定义,如果一个工作者的经验能够覆盖那些没有被覆盖的技术术语,那么该工作者会被赋予较高的多样性值.多样性通过提高选择出来的工作者的经验包含的技术术语的覆盖度,从而提高对于测试需求的覆盖性和缺陷的检测率.

3.3 选择算法

我们选择方法的目标是最大化主动性、相关性和多样性.基于对以上 3 个方面度量的定义,主动性和相关性对于每个工作者是独立的,因此最大化主动性只需要计算每个工作者的主动性指标,然后选择具有最大的主动性的工作者.类似的,最大化相关性只需要计算每个工作者的相关性指标,并且选择具有最大相关性的工作者.然而,对于某个工作者的多样性的衡量依赖于已经选择出来的工作者.也就是说,为了计算某个工作者的多样性,我们需要知道哪些工作者已经被选择出来了.

要保证 3 个方面都最大化是特别困难的,需要复杂的算法和很大的时间开销.因此,我们设计了基于贪心策略的选择算法.该算法希望平衡这 3 个方面,得到一种相对较优的选择方案.为了优化这 3 个方面,我们采用一个排序函数为每个候选工作者赋予一个分数,在每个迭代中,有着最大分数的工作者作为最佳工作者被选中.选择出来的最佳工作者和已经选择出来的工作者不仅有较大差异,还有着较大的主动性和相关性的.

排序函数如下所示:

$$w^*=\arg\max_{w_i\in W}\{\theta_1\times I(w_i)+\theta_2\times R(tk\mid w_i)+\theta_3\times D(w_i,S)\},\ \text{s.t.}\ \sum_l\theta_l=1 \tag{6}$$

其中,$I(w_i)$是工作者的主动性,$R(tk|w_i)$是工作者和测试任务 tk 之间的相关性,$D(w_i,S)$是工作者相对于已经选择出来的工作者集合 S 的多样性.θ_1 是主动性的权重参数,θ_2 是相关性的权重参数,θ_3 是多样性的权重参数,这 3 个参数中,取值越高,说明其对应的方面越重要;3 个权重参数的取值范围均为[0,1],且满足 3 个参数和为 1 的限制.排序函数中的权重可以根据各个方面的重要性进行决定,既可以基于专家经验确定权重,也可以通过验证集得到并用于其他项目中(详见第 4.3 节).注意:在计算排序函数时,这 3 个方面的度量值需要进行最小-最大归一化,使他们处于同一区间内.

当新的方面需要考虑时,该排序函数可以很容易地扩展,添加其他方面的度量值.因此,我们的选择算法是灵活的.

整个选择过程如下:在每次迭代中,排序函数给予集合 W 中的每个候选众测工作者一个分数,具有最大分数的工作者被选中,并且添加到已经选择出来的集合 S 中.整个选择过程重复进行,直到选择出来的人员数目满足要求.算法 1 中给出了详细的步骤.

算法 1. 选择算法.

输入:W:候选众测工作者;tk:测试任务;k:选择众测工作者的个数;$\theta_1,\theta_2,\theta_3$;

1. $S\leftarrow\varnothing$
2. While $|S|<k$ do
3. $w^*=\arg\max_{w_i\in W}\{\theta_1\times I(w_i)+\theta_2\times R(tk\mid w_i)+\theta_3\times D(w_i,S)\}$, s.t. $\sum_l\theta_l=1$
4. $W\leftarrow W\backslash\{w^*\}$
5. $S\leftarrow S\cup\{w^*\}$
6. end while
7. 返回 S

4 方法验证

基于第 1.3 节介绍的百度众测数据集进行实验,我们对第 3 节提出的众测工作者选择方法进行验证.

从两个维度验证:一是与基线方法进行比较,说明方法的有效性;二是验证 3 个方面的必要性.主要回答以下

4 个研究问题.

研究问题 1:与基线方法相比,本文方法在真实数据集上的效果如何?

研究问题 2:当只考虑主动性、相关性和多样性中的一个或者两个方面时,方法的效果如何?

研究问题 3:主动性、相关性和多样性这 3 个方面是如何相互补充的?

研究问题 4:考虑实际应用场景,方法的效果如何?

4.1 评价指标

为了评价工作者选择方法的有效性,我们从两个方面来评价工作者选择方法:首先,选择工作者的最终目标是希望被选择的工作者可以在众测任务中成功的发现缺陷,因此我们利用缺陷检测率度量一组选择的工作者能够发现缺陷的比例;此外,测试需求覆盖度也是测试活动的重要目标,它能够直接影响测试的有效性,具体来讲,众测发布者发布的测试需求希望找到与其经验相关的人,因为如果某个需求被忽略,那么这个需求所对应的缺陷也可能被遗漏,因此,我们通过测试需求关键点覆盖率来度量测试需求被选择工作者经验覆盖的比例.

- 缺陷检测率

虽然我们无法获得众测任务中理想的全部缺陷,但可以基于已经完成的众测任务,获得已经完成任务所发现的缺陷和发现缺陷的工作者,并基于此判断工作者选择方法是否能从候选工作者集合中选择出真正发现缺陷的正确工作者,并且计算这部分被选出的工作者检出的缺陷占总共被检出缺陷的比例.

对于某工作者集合,缺陷检测率定义为所选工作者检测到的缺陷占所有历史发现缺陷的比例:

$$\text{缺陷检测率}=\frac{\text{选择出的工作者发现缺陷的个数}}{\text{缺陷总数}}\times 100\% .$$

- 测试需求关键点的工作者覆盖度

与测试用例覆盖度度量不同,众测中无法具体控制某条测试需求是否被测试.我们试图度量是否选择了经验与测试需求相似、并且覆盖所有测试需求关键点的工作者,以此保证工作者有足够的经验去发现缺陷.

对于某个测试需求,某工作者集合对需求关键点的覆盖度定义为这些工作者的历史测试报告涉及的技术术语占测试需求中所有技术术语的比例:

$$\text{测试需求关键点的工作者覆盖度}=\frac{\text{测试需求的关键词被选择出的工作者覆盖的个数}}{\text{测试需求中关键词的总数}}\times 100\% .$$

4.2 基线方法

已有的研究中,并没有直接可以用来为众测环境下的测试任务选择一组合适工作者的研究,我们根据其他领域的人员选择方法[6−8],设置了 3 种简单直接的基线方法.

- 随机选择方法:该方法从候选工作者集合中随机选择工作者.该基线模拟的是目前众测平台上采用的自取模式(详见第 1.1 节),其中,工作者可以自由选择测试任务来执行.为了避免随机性的影响,我们将该基线重复进行 10 次,并且选择效果最好的那次结果作为该基线的最终结果;
- 活跃用户方法:在真实的众测平台上,有些测试任务的发布者通常希望邀请一些活跃用户来参与任务.该基线对应这个场景,选择过去提交报告最多的工作者.这与只用主动性指标的排序方法类似;
- 信息检索方法:另一个常用的方法是运用信息检索的方式搜索对测试任务熟悉的用户来参与任务.该场景下,将测试任务作为检索,将工作者历史提交的测试报告作为文档.有着最高相似性的工作者被选择出来.这与只用相关性指标的排序方法类似.

4.3 实验设置

为了评价模型效果,我们进行线下模拟实验,基于包含已经完成测试任务的工业数据集合,实验包括如下两个步骤:

(1) 调整确定模型参数

对于本文方法,需要调整 3 个参数:主动性的权重θ_1、相关性的权重θ_2以及多样性的权重θ_3.由于公式(6)中

限定这 3 个参数的和为 1,因此我们只需要调整其中两个参数即可.在本文实验中,我们只调整θ_1和θ_2,θ_3可以通过 $1-\theta_1-\theta_2$ 的方式获得.

我们从整个实验数据集中抽样得到一个子集进行参数的调整,该子集包含 25%的测试任务(共 12 个).θ_1和θ_2被限定在[0.0,1.0]的范围内,每隔 0.05 作为一个实验区间,并且满足 3 个参数的和为 1.我们用留一交叉检验的方式确定参数的值:在每次实验中,一个测试任务作为验证集,其余测试任务作为训练集,重复进行 12 次.然后,将 12 次实验的结果求平均值.我们将集合上效果最好的参数作为最终参数(θ_1=0.05,θ_2=0.4,θ_3=0.55),用于后续实验进行方法的验证.

对于输入参数 k(详见第 3 节),我们研究 k 取 5~15 的所有整数值来综合评估方法的性能.

(2) 验证模型性能

本文设计两个实验验证模型性能:1) 交叉验证实验,采用留一交叉检验的方式进行实验验证,留一交叉验证的方法被使用在很多软件工程任务的验证中,如缺陷预测[27]、软件版本发布计划[28];2) 模拟实际应用场景的线下实验,考虑众测任务的顺序和历史数据积累过程对模型的影响.

- 交叉验证实验

具体来讲,对于所有 46 个已经完成的测试任务,选择一个测试任务作为测试集,其余的测试任务作为训练集,通过训练集作为历史数据来计算主动性、相关性和多样性的指标,然后判断在测试集上方法是否选择正确工作者,按照第 4.1 节计算评价指标.重复实验 46 次,将 46 个结果取平均值,作为最终的结果.

- 模拟实际应用场景的线下实验

我们按照众测任务完成时间,把 46 个测试任务按照先后顺序分配到 5 个时间桶中,这里,每个时间桶包含基本相等数量的众测任务(时间桶 0~4 分别包含 10,9,9,9,9 个测试任务),由此,5 个桶之间产生 4 个时间点.我们使用在某一时间点之前的测试任务作为训练集,该时间桶中的测试任务作为测试集,分别为 5 个时间桶中的众测任务推荐工作者,并评价推荐结果,在每个时间桶中计算平均的评价指标.

4.4 结果和分析

本节给出实验结果,回答前面提出的 4 个研究问题.

4.4.1 与基线方法相比,本文方法在真实数据集上的效果

图 7 给出本文方法以及 3 条基线方法的测试需求关键点覆盖度和缺陷检测率.可以很容易地看出,随机选择方法的效果最差,这表明需要设计一种专门的算法来为众测环境下的测试任务选择工作者.与 3 条基线方法相比,本文方法可以达到最好的测试需求关键点覆盖度和缺陷检测率.当 k 为 15 时(选择 15 个工作者),我们方法的测试需求关键点覆盖度(62%)比随机选择方法高出 10%,缺陷检测率(45%)比随机选择方法高出 20%.与性能排名第 2 的信息检索方法相比,本文方法在测试需求关键点覆盖度上提高了 7%,在缺陷检测率上提高了 10%.

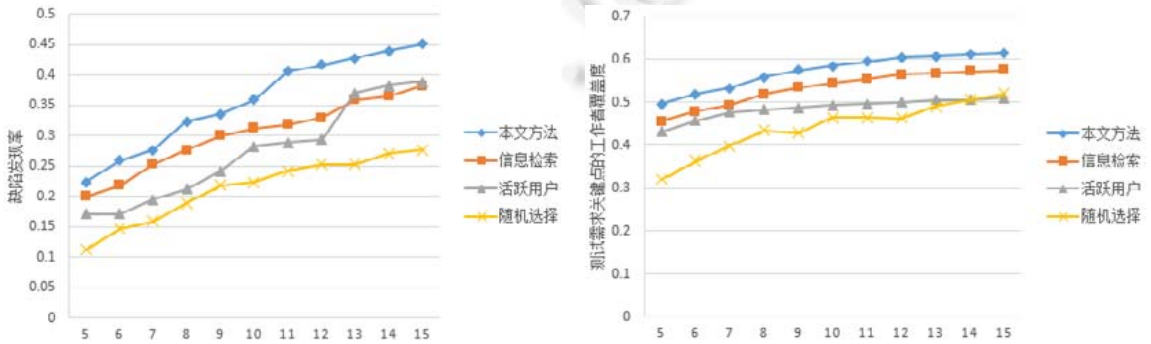


Fig.7 Comparison of result with baseline approaches

图 7 与基线的对比结果

进一步地,信息检索方法得到的测试需求关键点覆盖度和缺陷检测率比活跃用户方法更高,这说明工作者的经验比工作者参与任务的数目对于人员选择更有用.

随着 k 的增加,测试需求关键点覆盖度和缺陷检测率都增加,这是因为随着更多的工作者被选择到参与任务中,更多的测试需求关键点被覆盖,并且更多的缺陷可能被检测到.为了达到相同的测试需求关键点覆盖度或者缺陷检测率,本文方法比基线方法需要更少的工作者.这可以更大程度地利用工作者,节省众测成本,对于众测任务是很有价值的.

4.4.2 本文方法在只考虑主动性、相关性和多样性中的一个或者两个方面时的效果

对于选择 3 个方面中的两个,参数仍然通过与第 4.3 节类似的交叉检验方法得到.在研究问题 1 的结果中,我们已经涉及了只考虑主动性和相关性的结果.这里我们主要考虑其他的情况.

图 8 显示所有 3 个方面的结合能得到最好的结果.删掉任何一个方面都会导致测试需求关键点覆盖度和缺陷检测率的下降.这表明:当选择工作者来执行某个测试任务时,每个方面都应该被考虑.

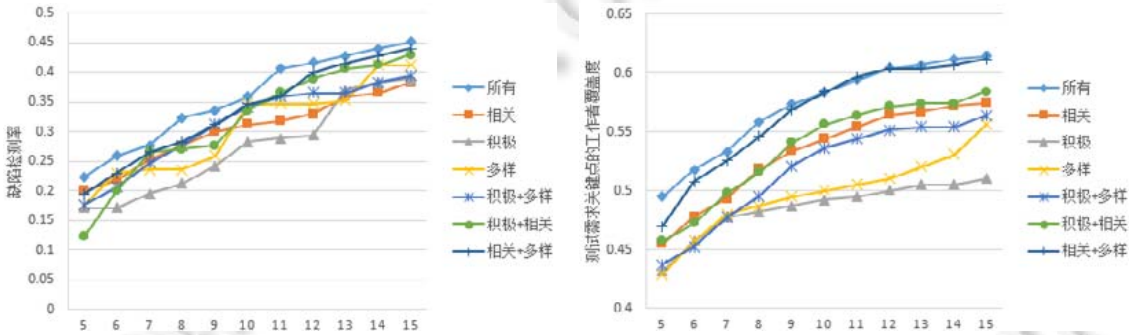


Fig.8 Performance using one or two of the three aspects

图 8 仅利用一个或两个方面的性能比较

此外,当考虑任何两个方面时,测试需求关键点覆盖度和缺陷检测率都比只考虑一个方面要好.当考虑两个方面时,相关性和多样性的结合能得到最好的效果.这比较容易理解,因为相关性和多样性单独使用带来的效果是只考虑一个方面时最好的.虽然主动性的贡献看起来不是很大,但是在其他方面中,增加主动性也会带来方法效果的提升.

4.4.3 主动性、相关性和多样性的相互补充

为了回答这个问题,我们研究由这 3 个方面选择出来的工作者的重叠度.

对于每个测试任务,我们基于每个单独的方面选择得到一个工作者的子集(每个子集 15 人).对于每个子集,我们得到在该任务中检测出来缺陷的工作者(简称缺陷检测者),并用于后续研究.为了检查主动性、相关性、多样性是如何相互补充的,我们研究基于 3 个不同方面得到的缺陷检测者的重叠度.详细来说,统计只被一个方面选择得到的缺陷检测者(称为不重合组)以及被多于一个方面选择得到的缺陷检测者(称为重合组).然后,对于每个测试任务,计算两组不同类型的缺陷检测者的比例,图 9 给出详细的结果.

我们发现在多于 90%的测试任务中,多于 50%的缺陷检测者只被一个方面选中.这表明:主动性、相关性、多样性是相互补充的,去掉任何一个方面都会降低总体的性能.

4.4.4 本文方法在考虑实际应用场景的效果

在实际应用场景中,首先,众测任务有着时间先后的关系,利用交叉验证的方法忽视了测试任务的先后.另外,在实际应用场景中,历史的数据是不断积累逐渐增多的,起初数据可能不是很充分.由于以上实际应用场景中的因素,我们需要在线下模拟实际应用场景中的情况,进一步确认我们方法的效果.

如第 4.3 节中所述,将数据集合按时间先后顺序分成 5 个时间桶,使用在某一时刻之前的测试任务作为训练集,该时间桶中的测试任务作为测试集,分别为 5 个时间桶中的众测任务推荐工作者,并评价推荐结果,在每个时

间桶中计算平均的评价指标.

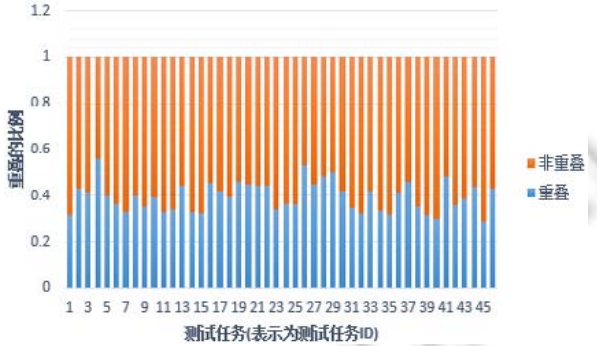


Fig.9 Overlap of selected workers by the three aspects

图 9 3 个方面选择出的工作者的重合率

如图 10 所示:历史数据的多少确实对模型的性能存在影响.在历史数据积累较少时,模型的性能相对较低.这是因为模型是基于历史数据的.实际上,在历史数据少的情况下,也没有更好的方法来预测.然而随着历史数据的增加,本文方法的性能迅速趋近于前面留一交叉验证实验的结果(留一交叉验证利用了更多的历史数据,如前文所述,除了该众测任务的数据,其余数据全部作为训练集合).由此可知:本文方法能有效利用历史数据,并且迅速达到较优的预测结果.

此外,在考虑测试任务的先后顺序后,我们发现测试任务的先后顺序对于本文方法的性能并没有明显的影响,即在考虑众测任务的时间顺序的情况下,模型随着时间不断接近交叉验证结果.这说明用户随着时间的历史经验积累是比较稳定、持续的过程(如果用户性质随着时间发生剧烈变化,那么后一个时间点上的性能有可能相比前一个时间点不增加,甚至倒退).如图 10 所示:不同时间点的性能曲线,随着时间是一个比较平稳并比较快速的收敛过程,并没有出现不增长或者倒退,说明了用户的历史经验积累是稳定的,并不存在剧烈的波动变化.

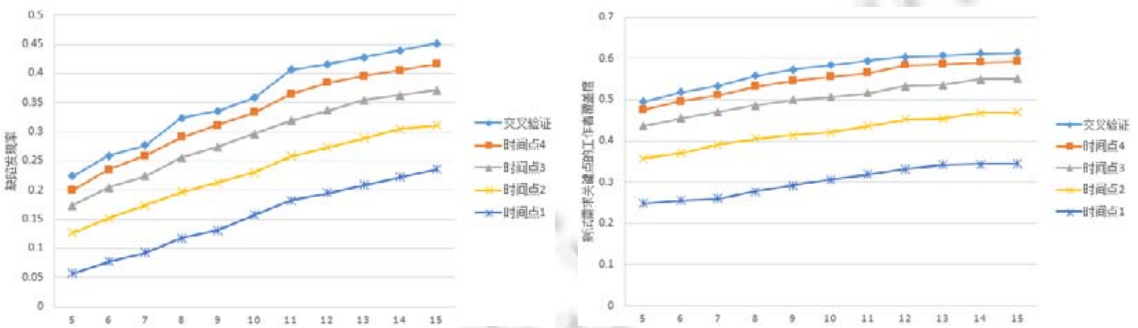


Fig.10 Performance of our approach in different time points

图 10 不同时间点上方法的性能对比

5 讨 论

5.1 *k*的设置

众所周知,众测经常涉及很多的工作者来执行某测试任务.读者可能会质疑实验中的 k 设置的比较小.这是因为第 1.3 节提到:本文用到的实验数据集中,对于某个测试任务,平均有 37 个工作者提交测试报告,平均 15 个工作者检测到缺陷.真实的众测平台上的真实场景,使得我们采用这样的实验设置.进一步的,我们方法的目的是选择尽可能少的工作者,检测到尽可能多的缺陷.这能够在保持测试产出不变的情况下,减少测试任务发布者

的开销,使他们更愿意在众测平台上发布任务,从而促进众测平台的繁荣.

此外,实验验证结果显示:当选择同样数目的工作者时,相比其他方法,本文方法能够检测到更多的缺陷.因此认为当 k 变大时,本文方法仍然可以得到较好的性能.

5.2 工作者选择策略

读者可能认为,对于测试需求中的每个技术术语,工作者选择方法需要选择多于一个工作者来进行覆盖.这是因为在推送模式下,不是每个工作者都会接受邀请并且执行测试任务的.实际上,在本文方法中,对于测试需求中的每个技术术语,确实能够选择到多于一个的工作者.第 1 个原因是,本文方法同时考虑其他方面,例如主动性和相关性,这会帮助选择到覆盖同一技术方面的工作者;第 2 个原因是,当实现多样性时,我们使用概率相关的度量,该度量在选择人员时,只是暂时降低已经满足的技术方面的概率,并不是完全去掉这些已经满足的技术方面.这样的处理使得本文方法可以为某一技术方面选择多于一个的工作者.

5.3 方法对新老工作者倾向性分析

由于本文方法基于工作者在众测平台的历史数据,为众测任务选取工作者,那么本文方法是否倾向于为众测平台工作时间较长的老工作者,而忽视刚进入到众测平台、历史数据不多的新工作者呢?首先,在第 4.4 节的研究问题 1 中,通过比较本文方法和单纯选取老用户方法的性能,结果显示本文方法性能明显优于单纯选取老用户的方法,其中,老用户就是平台上工作时间最长的老工作者集合(类似基线方法中的活跃用户方法).这个结果说明,单纯选择老用户并不能提高缺陷发现率.进一步的,我们对本文方法的所选工作者进行深入分析,如图 11 所示,在比较了本文方法和老用户方法在 46 个测试任务中,当 k=15 时所选出的工作者提交过测试报告的分布.我们可以发现:本文方法并不倾向选出提交测试报告最多的那一部分老工作者,而主要集中在比较有经验、在不同的任务中又有相应专业知识的工作者.

众测平台上有部分新工作者,即刚进入平台、还没有历史数据的工作者.由于本文方法是基于工作者的历史进行的建模和推荐,所以不能为他们推荐任务.后续工作会考虑基于工作者填写的属性信息为他们进行推荐,从而增加他们参与任务的积极性.新工作者可以自己搜索在线正在发布的任务去参与,待工作者有历史数据后,再由本文方法为其推荐众测任务.

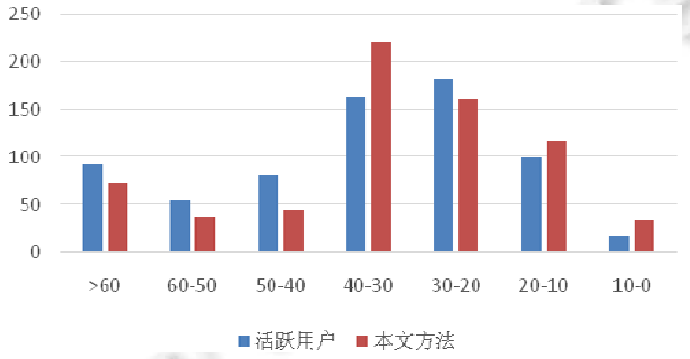


Fig.11 Number of test reports of workers selected by our approach

图 11 方法选择的工作者历史提交测试报告的数目

5.4 对有效性的威胁

对外部有效性的威胁主要是本研究的通用性.

- 首先,实验数据包含了从中国最大的众测平台之一收集到的 46 个测试任务,我们不能完全保证在别的实验环境下也能得到同样的结果.然而,我们的数据集相对较大,而且其中包含各个领域的项目(例如音乐、工具、游戏等),这帮助我们在一定程度上减轻了这个威胁;
- 其次,本研究中的所有众测报告都是用中文写的,我们不能保证在其他语言的众测项目上得到同样的

结果.然而,因为我们没有进行语义分析,只是对文本进行分词,用词作为标记来进行建模,因此,该威胁得到极大的减轻.

对于内部有效性的威胁,方法中涉及的 3 个参数θ_1,θ_2和θ_3可能影响我们的实验结果.为了控制该方面的威胁,我们抽样得到部分数据,并用交叉检验的方式来选择可以得到最好性能的参数集合.此外,我们度量主动性是假设主动性在一定的时间窗口内是保持稳定的,即,工作者的主动性在一定时间内不会随时间发生剧烈变化.然而在很长的时间范围内,工作者的主动性确实可能发生转变,比如有些原来主动的工作者变得不主动了、原来不主动的工作者逐渐主动.我们计划在未来的工作中,研究时间对于主动性的影响.

对于构造有效性的威胁,本研究考虑主动性、相关性和多样性这 3 个方面.这 3 个方面是从不同的角度设计的:单个工作者的独特性、一组工作者的多样性、他们的经验和任务的关系等.然而,其他方面也可能会影响测试需求的覆盖度和缺陷的检测率.为了解决这个威胁,需要对其他方面进行研究和建模.

6 相关工作

6.1 众 测

随着工业环境下众测模式的快速发展,众测也吸引着越来越多学术界的研究者.众测通过将众包的概念引入测试领域,能够帮助集中的软件开发和测试工程师发现缺陷[9−11].

众测相关的研究主要分为两个方向.

- 用众测这种新兴的测试模式来辅助解决传统软件测试中的问题.Pastore 等人[12]研究是否能够用众测解决 oracle 问题,他们将反映当前程序的行为组织成断言,并将这些断言作为众测任务发布到众测平台上,工作者需要评估这些断言的正确性.Liu 等人[13]将众测应用到可用性测试的研究中,他们通过经验研究,发现众测对于可用性测试的适用性和价值.Nebeling 等人[14]开发了一个工具包,可以支持众测模式下的网页测试,该工具包不仅能够快速地招募大量的工作者,还能够在不同的条件下评估网站.
- 关注如何解决众测环境下产生的新问题.Feng 等人[1]提出一种方法对众测环境下的测试报告进行排序,他们综合运用多样性策略和风险策略,动态选择测试报告进行检查.Wang 等人[2,3]提出的方法可以从大量的测试报告中分类得到真正含有缺陷的测试报告.Tung 等人[15]研究如何更有效地为协同的测试任务分配工作者,他们将该问题建模为线性规划问题.

我们的工作关注的是众测环境下新产生的问题,也就是如何为众测任务选择一组合适的工作者.根据我们的调研,这是第 1 个此种类型的工作.

6.2 缺陷检测和测试覆盖

缺陷检测是软件测试活动中的一个重要目标[16].在传统的测试中,很多方面被认为会影响缺陷检测,例如测试代码质量、测试用例选择、测试充分性准则等.Athanasiou 等人[17]通过反映测试代码质量的 3 个方面——完整性、有效性和可维护性来评估测试代码质量.结果显示,测试代码质量和缺陷检测率之间存在显著相关性.Rothermel 等人[18]给出几种方法对测试用例进行排序,结果表明,测试用例排序可以显著地提升缺陷检测率.Zhou 等人[19]研究哪些测试的充分性准则对于测试 java 数据库应用是最适合的,结果发现,语句覆盖或者分支覆盖是最有效的.不同于之前的工作,我们的研究聚焦人员相关的因素,这是众测环境下新产生的,并且对于众测是很关键的.

覆盖性是软件测试活动的另一个指标,已有的研究从多个不同的角度研究覆盖性,例如测试用例的覆盖、测试数据的覆盖等.Gopinath 等人[20]将覆盖性准则作为测试集的质量指标.Leon 等人[21]基于多元可视化技术提出一种新的测试数据选择方法.Mondal 等人[22]在几个真实的测试用例选择的案例研究中,比较了代码覆盖和测试用例多样化的关系.本文方法关乎测试需求的的覆盖性,我们从工作者的角度研究覆盖性,并且用人员经验的多样性来提高这个覆盖度.此外,我们不仅从多样性的角度研究测试需求的覆盖性,还考虑可能影响覆盖性的其他方面.

6.3 人员选择

软件开发已经成为一项越来越开放的活动,其中经常涉及来自开放的大众工作者.为某个软件开发任务推荐合适的工作者正变得越来越重要.有很多相关工作关注为各种各样的软件开发任务选择合适的工作者,例如推荐缺陷修复人、为新成员推荐导师、推荐某个领域的专家等.

Jeong 等人[23]研究如何为一个新的缺陷报告推荐可能修复该缺陷的开发者,他们引入了基于马尔科夫链的图模型来建模缺陷报告在人员之间的转移历史.Tamrawi 等人[24]提出一种新的方法进行缺陷修复人的推荐,他们通过缓存开发者的缺陷修复历史以及基于模糊集的方法.Canfora 等人[25]通过挖掘邮件列表和版本控制系统的数据,为开源社区中的新成员推荐导师.Ma 等人[26]提出了使用专长的概念,并且通过评估专家推荐的效果说明该概念的可行性.

已有的研究要么只是推荐一个工作者,要么假设推荐的一组工作者是相互独立的.然而在众测环境下,我们需要选择一组工作者,并且工作者之间是互相依赖的,因为他们共同完成一个团队工作.

7 结束语

由于众测环境下的工作者分布在不同地域,有着不同的测试经验,哪些工作者执行某一测试任务会大大影响测试需求关键点覆盖度和缺陷检测率.本文识别了众测环境下影响工作者缺陷发现的 3 个方面,分别是主动性、相关性和多样性,并且提出一种众测环境下的人员选择方法.该方法能够同时考虑主动性、相关性和多样性,从而提高测试需求关键点覆盖度和缺陷检测率.我们基于百度众测的真实数据验证了本方法,结果显示了方法的有效性,当选择 15 个工作者时,本文方法可以得到 62%的测试需求关键点覆盖度和 45%的缺陷检测率,优于基线方法.

在未来的工作中,我们计划研究其他可能影响众测环境下缺陷发现的方面,考虑工作者的主动性、专业方向是否会随时间发生转变.并且,我们一直和百度保持着密切的合作,计划将本文方法部署到线上环境,从而更好地验证方法在实际环境下的有效性.

References:

[1] Feng Y, Chen Z, Jones JA, Fang C, Xu B. Test report prioritization to assist crowdsourced testing. In: Proc. of the 2015 10th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2015). New York: ACM Press, 2015. 225−236.

[2] Wang J, Cui Q, Wang Q, Wang S. Towards effectively test report classification to assist crowdsourced testing. In: Proc. of the 10th ACM/IEEE Int'l Symp. on Empirical Software Engineering and Measurement (ESEM 2016). 2016. 6:1−6:10.

[3] Wang J, Wang S, Cui Q, Wang Q. Local-Based active classification of test report to assist crowdsourced testing. In: Proc. of the 31st IEEE/ACM Int'l Conf. on Automated Software Engineering (ASE 2016). 2016. 190−201.

[4] Hochba DS. Approximation algorithms for NP-hard problems. ACM Sigact News, 1997,28(2):40−52.

[5] Hiemstra D. Using Language Models for Information retrieval. Taaluitgeverij Neslia Paniculata, 2001.

[6] Canfora G, Di Penta M, Oliveto R, Panichella S. Who is going to mentor newcomers in open source projects? In: Proc. of the ACM SIGSOFT 20th Int'l Symp. on the Foundations of Software Engineering (ESEC/FSE 2012). New York: ACM Press, 2012. 44:1−44:11.

[7] Dror G, Koren Y, Maarek Y, Szpektor I. I want to answer; Who has a question? Yahoo! Answers recommender system. In: Proc. of the 17th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD 2011). New York: ACM Press, 2011. 1109−1117.

[8] Zhao Z, Wei F, Zhou M, Chen W, Ng W. Crowd-Selection query processing in crowdsourcing databases: Atask-driven approach. In: Proc. of the 18th Int'l Conf. on Extending Database Technology (EDBT 2015). 2015. 397−408.

[9] Chen Z, Luo B. Quasi-Crowdsourcing testing for educational projects. In: Companion Proc. of the 36th Int'l Conf. on Software Engineering (ICSE Companion 2014). New York: ACM Press, 2014. 272−275.

[10] Mao K, Capra L, Harman M, Jia Y. A survey of theuse of crowdsourcing in software engineering. Technical Report. 2015.

[11] Feng JH, Li GL, Feng JH. A survey on crowdsourcing. Chinese Journal on Computers, 2015,38(9):1713−1726 (in Chinese with English abstract).

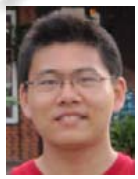
[12] Pastore F, Mariani L, Fraser G. Crowd oracles: Can the crowd solve the oracle problem? In: Proc. of the 2013 IEEE 6th Int'l Conf. on Software Testing, Verification and Validation (ICST 2013). 2013. 342−351.

[13] Liu D, Bias RG, Lease M, Kuipers R. Crowdsourcing for usability testing. Proc. of the American Society for Information Science and Technology, 2012,49(1):1−10.
[14] Nebeling M, Speicher M, Grossniklaus M, Norrie MC. Crowdsourced Web Site Evaluation with Crowdstudy. Springer-Verlag, 2012.
[15] Tung YH, Tseng SS. A novel approach to collaborative testing in a crowdsourcing environment. Journal of Systems and Software, 2013,86(8):2143−2153.
[16] Bertolino A. Software testing research: Achievements, challenges, dreams. In: Proc. of the 2007 Future of Software Engineering (FOSE 2007). Washington: IEEE Computer Society, 2007. 85−103.
[17] Athanasiou D, Nugroho A, Visser J, Zaidman A. Test code quality and its relation to issue handling performance. IEEE Trans. on Software Engineering, 2014,40(11):1100−1125.
[18] Rothermel G, Untch RH, Chu C, Harrold MJ. Test case prioritization: An empirical study. In: Proc. of the IEEE Int'l Conf. on Software Maintenance (ICSM'99). 1999. 179−188.
[19] Zhou C, Frankl P. Empirical studies on test effectiveness for database applications. In: Proc. of the 2012 IEEE 5th Int'l Conf. on Software Testing, Verification and validation (ICST 2012). 2012. 61−70.
[20] Gopinath R, Jensen C, Groce A. Code coveragefor suite evaluation by developers. In: Proc. of the 36th Int'l Conf. on Software Engineering (ICSE 2014). New York: ACM Press, 2014. 72−82.
[21] Leon D, Podgurski A, White LJ. Multi variatevisualization in observation-based testing. In: Proc. of the 22nd Int'l Conf. on Software Engineering (ICSE 2000). New York: ACM Press, 2000. 116−125.
[22] Mondal D, Hemmati H, Durocher S. Exploring testsuite diversification and code coverage in multi-objectivetest case selection. In: Proc. of the 2015 IEEE 8th Int'l Conf. on Software Testing, Verification and Validation (ICST 2015). 2015. 1−10.
[23] Jeong G, Kim S, Zimmermann T. Improving bugtriage with bug tossing graphs. In: Proc. of the 7th Joint Meeting of the European Software Engineering Conf. and the ACM SIGSOFT Symp. on The Foundations of Software Engineering (ESEC/FSE 2009). New York: ACM Press, 2009. 111−120.
[24] Tamrawi A, Nguyen TT, Al-Kofahi JM, Nguyen TN. Fuzzy set and cache-based approach for bug triaging. In: Proc. of the 19th ACM SIGSOFT Symp. and the 13th European Conf. on Foundations of Software Engineering (ESEC/FSE 2011). New York: ACM Press, 2011. 365−375.
[25] Ma D, Schuler D, Zimmermann T, Sillito J. Expertrecommendation with usage expertise. In: Proc. of the IEEE Int'l Conf. on Software Maintenance (ICSM 2009). 2009. 535−538.
[26] Yan X, Guo J, Lan Y, Cheng X. A biterm topic model for short texts. In: Proc. of the 22nd Int'l Conf. on World Wide Web (WWW 2013). New York: ACM Press, 2013. 1445−1456.
[27] Wang S, Liu T, Tan L. Automatically learning semantic features for defect prediction. In: Proc. of the 38th Int'l Conf. on Software Engineering (ICSE 2016). New York: ACM Press, 2016. 297−308.
[28] Villarroel L, Bavota G, Russo B, Oliveto R, Penta M. Release planning of mobile apps based on user reviews. In: Proc. of the 38th Int'l Conf. on Software Engineering (ICSE 2016). New York: ACM Press, 2016. 14−24.
[29] Zhou M, Mockus A. What make long term contributors: Willingness and opportunity in OSS community. In: Proc. of the 34th Int'l Conf. on Software Engineering (ICSE). 2012. 518−528.

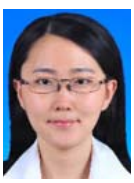
附中文参考文献:

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

崔强(1985－),男,辽宁抚顺人,博士生,CCF学生会员,主要研究领域为众测,推荐算法.



谢淼(1988－),男,博士,工程师,主要研究领域为数据挖掘,软件工程.



王俊杰(1987－),女,博士,副研究员,主要研究领域为缺陷预测,经验软件工程,众测.

王青(1964－),女,博士,研究员,博士生导师,CCF高级会员,主要研究领域为软件过程技术,需求工程,软件质量与管理.