DACOS——代码异味人工标注数据集

Himesh Nandani、Mootez Saad、Tushar Sharma达尔豪斯大学哈利法克斯，加拿大{himesh.nandani, mootez, tushar}@dal.ca

arXiv:2303.08729v1 [cs.SE] 15 Mar 2023

摘要——**研究人员运用机器学习技术进行代码异味检测，以应对许多代码异味所具有的主观性。这类方法需要大型人工标注数据集，以用于训练和基准测试。现有文献提供了一些数据集，但它们规模较小；更重要的是，这些数据集并未专注于主观代码片段。本文介绍了**Dacos**，这是一个人工标注数据集，包含**10,267** 条针对** 5,192 **个代码片段的标注。该数据集涵盖不同粒度的三类代码异味——**多面抽象、复杂方法**和过长参数列表。数据集分两个阶段创建。第一阶段通过确定用于检测异味的度量指标阈值，帮助我们识别可能具有主观性的代码片段。第二阶段则收集这些可能具有主观性的代码片段的标注。我们还提供了扩展数据集**DacosX**，该数据集利用第一阶段确定的阈值，纳入了明确无异味和明确有异味的代码片段。我们开发了**Tagman**这一 Web 应用程序，帮助标注人员逐一查看和标记代码片段，并记录提供的标注。我们公开了这些数据集和 Web 应用程序。该数据集将帮助从事异味检测技术研究的人员构建相关的上下文感知机器学习模型。**

I. 引言

代码异味是糟糕的设计和实现的症状 [1]。现有文献表明，代码异味会对可维护性 [2], [3], 、开发工作量[4], [5],和可靠性 [6]等质量属性产生负面影响[9] 。鉴于其重要性，软件工程界投入了大量精力研究与代码异味相关的各个方面，例如其特征、影响、成因和检测机制 [10]。

许多代码异味具有主观性 [10] ，也就是说，某段代码在一种上下文中可能存在异味，但类似的代码片段在另一种上下文中却未必会被认为有异味。上下文包括所使用的编程语言、开发团队的经验，以及组织所遵循的质量相关实践。一个简单的例子是代码异味的主观性：例如，一个有 80 行代码的方法。根据上下文，某些开发者可能会认为它是过长的方法；其他人则可能不会将其归类为过长的方法。不过，一个有 500 行代码的方法无疑会被所有开发者视为过长的方法。

目前，大多数常用工具都使用度量指标和启发式方法来识别代码异味 [10]。人们常常认为，由于异味具有主观性，因此无法提出一套普遍认可的度量阈值，用于在所有上下文中将代码片段分类为有异味或无异味代码。 [10],[11] 为克服异味主观性带来的挑战，研究人员提出了使用机器学习技术检测异味的方法 [11]–[16]。此类方法依赖代码异味数据集（理想情况下应为人工标注），以训练机器学习模型。然而，现有数据集在多个方面都存在不足。首先，文献中只有少数几个数据集，例如 LANDFILL [17]。其次，现有数据集所含的已标注样本数量较少；例如，LANDFILL 仅提供了 243 个代码片段的标注。已标注样本数量较少的数据集，对于以合理的准确率训练最先进的深度学习模型帮助有限。再者，现有代码异味数据集并未过滤掉明确无异味或有异味的代码片段。例如，只有寥寥几行（比如三行）代码的片段不可能存在过长方法；类似地，代码行数非常多（例如，200 行）的片段肯定存在过长方法异味。鉴于异味数据集的价值体现在其对主观性的捕捉上，这些明确无异味或明确有异味的代码片段会降低数据集的效用。最后，现有数据集对不同类型异味的支持有限；例如，LANDFILL 仅提供了五种异味类型的已标注代码片段。鉴于标注代码片段需要投入大量精力，软件工程界需要补充现有异味数据集，涵盖其他受到积极研究的异味类型。

本文提供了一个代码异味人工标注数据集即 代码异味数据集 （DACOS）。为构建有效的数据集，我们通过剔除明显无异味或明显存在异味的代码片段，筛选出可能存在主观判断的代码片段。这种方法让我们能够将标注人员的精力更有效地用于真正需要他们提供意见的地方。该数据集包含三种代码异味的标注代码片段：多面抽象[18], [19],复杂方法 [20], 和过长参数列表 [1]。除了针对可能需要主观判断的代码片段的人工标注数据集，我们还提供了 DACOSX 数据集，其中包含大量明显无异味或明显存在异味的代码片段。此外，我们开发了一个 Web 应用程序即TAGMAN，方便标注人员逐一查看代码片段，并指出其中是否存在代码异味。我们已公开提供 TAGMAN 的源代码[1] 。

[1]https://github.com/SMART-Dal/Tagman

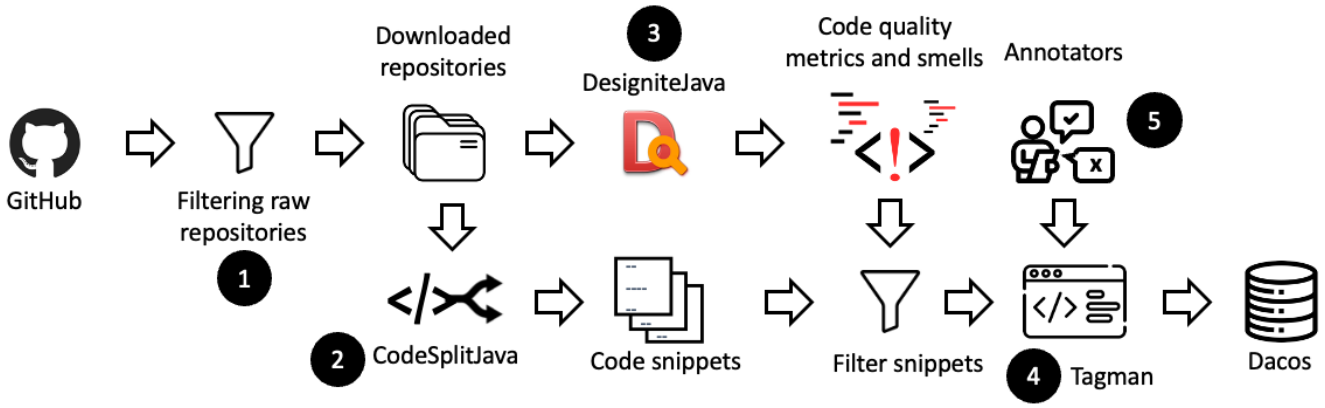

图 1. 数据集构建过程

我们为推动该领域的发展作出了以下贡献。

- 我们提供了 DACOS，这是一个经过人工标注的代码异味数据集，其中包含针对所研究代码异味的 10，267 条标注，涵盖 5，192 个代码片段。
- 我们还提供了 DACOSX，这是一个扩展数据集，包含 207，605 个代码片段，这些片段要么明确无异味，要么存在异味。这些数据集将帮助该领域的研究人员训练和验证其机器学习模型。
- 我们开发了可配置的 Web 应用程序 TAGMAN，便于进行代码异味标注。社区可以使用该应用程序开展类似的代码标注工作。

II. 数据集构建

图 1 概述了数据集构建过程。下面我们将详细说明各个步骤。

A. 下载代码仓库

在图 [redacted] 中，我们执行以下任务来识别并下载代码仓库。

- 我们使用 searchgithubrepo [21] Python软件包，该软件包使用 GITHUB GRAPHQLAPI [22] 筛选 GitHub代码仓库。
- 为了识别高质量的 Java代码仓库，我们选择星标数不少于 1.3 万且代码行数超过 1 万的代码仓库。
- 此外，我们会舍弃过去一年内未修改过的代码仓库。
- 此外，我们使用 QScored [23] 根据代码质量分数筛选代码仓库。QScored 会根据在不同粒度下检测到的异味分配加权质量分数。我们选择质量分数低于 10 的代码仓库（分数越高，质量越差）。
- 最后，应用筛选标准后，我们获得了十个代码仓库。我们下载了选定的代码仓库。

B. 将代码仓库划分为类和方法

我们需要将代码仓库拆分为独立的方法和类，以便 TAGMAN 能逐个向标注人员展示代码片段对标注人员逐个展示。我们使用 CODESPLITJAVA [24] 在步骤 [redacted] 中将每个代码仓库拆分为独立的方法和类。

C. 分析代码仓库

在步骤 [redacted] 中，我们在人工标注的第二阶段采用基于度量指标的筛选流程。我们使用 DESIGNITEJAVA[25] 计算代码质量度量指标。DESIGNITEJAVA 会计算各种代码质量度量指标并检测代码异味；它已在多项研究中得到应用 [26]–[30]。我们将详细说明在人工标注步骤中筛除非主观样本的流程。

D. Tagman

TAGMAN（如图 1 [redacted] 的 所示）是一款我们创建的基于Web的工具，旨在协助标注流程。图 2 展示了该应用的屏幕截图，其中包含一个代码片段，以及一个用于为该代码片段标注代码异味的选项。TAGMAN 的前端使用*Thymeleaf* 和 HTML/CSS 编写。该应用的后端使用*SpringBootJava*开发，数据存储在 MySQL数据库中。图 3 展示了数据库的架构。

在代码异味标注周期开始时，我们上传一个 CSV文件，其中包含所选 GitHub代码仓库的名称和 URL。TAGMAN 后端使用一组 Python脚本[2] 下载 GitHub代码仓库，将代码拆分为类文件和方法文件，并运行 DESIGNITEJAVA。

数据导入完成后，该工具便可开始接收标注。要开始标注会话，用户需要登录（或注册）应用程序。随后，用户会看到包含代码异味定义的说明，然后即可开始标注所展示的代码片段。

E. 人工标注

我们采用一种代码片段筛选机制，以识别相对于某种代码异味可能具有主观性的代码片段。这样做是为了提高最终数据集的有效性，只纳入针对可能具有主观性的代码片段进行的人工标注

[2]https://github.com/SMART-Dal/Tagman-scripts



图 2. TAGMAN 的标注用户界面

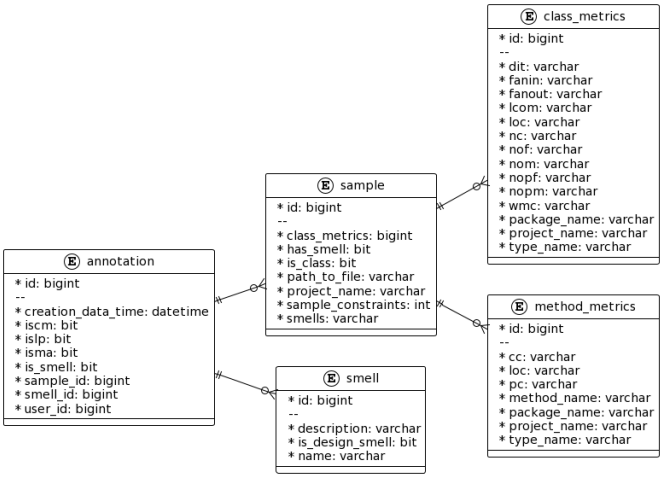

图 3. DACOS 数据库的架构

。此外，这种策略也有助于我们更好地利用标注人员的时间。可能具有主观性的代码片段，是指根据上下文可能被归类为正常或有异味的代码片段。其余未被识别为可能具有主观性的代码片段，则属于明确正常或明确有异味的代码片段。例如，圈复杂度 (CC)[31] 通常用于检测复杂方法 异味。如果 CC 很低（例如，CC=1），则该代码片段明确正常；类似地，如果某个方法的 CC 很高（例如，CC=30），则该代码片段明确有异味。我们将标注流程分为两个阶段。第一阶段，我们向标注人员展示所有代码片段，即，不进行任何筛选，以确定度量阈值，从而判断代码片段是否可能具有主观性。第二阶段则使用已确定的度量阈值，向标注人员展示筛选后的代码片段。

1) 第一阶段：在第一阶段，我们向标注人员展示未经筛选的代码片段。TAGMAN 一次向标注人员展示一个代码片段，并收集他们对所展示的代码片段是否存在代码异味的判断。我们将一个代码片段展示给随机选出的两名标注人员，并记录他们的回答。本阶段招募的志愿标注人员均为计算机科学专业研究生，他们选修了一门深入讲解代码异味的软件工程课程（2022 年夏季）。共有 110 名标注人员参与了本阶段。

我们根据第一阶段收集的回答，计算用于判定是否存在代码异味的度量指标的最小阈值和最大阈值。本阶段共收到 17，869 条回答。针对每种异味，我们分别计算识别出该异味时的最低度量指标值（t_l），以确定下限阈值。同样，我们提取未 识别出该异味时的最高度量指标值（t_h）。然后，计算识别出该异味的样本所对应度量指标值的标准差（sd）。最后，通过 $\max(m_l, t_l - sd)$ 和 $\min(m_h, t_h + sd)$ 得到主观片段识别的低阈值和高阈值。这里，m_l 和 m_h分别表示度量指标可能的最低值和最高值。表 I 总结了每种代码异味对应的质量度量指标，以及用于识别主观片段的阈值。例如，对于圈复杂度度量指标，将数值四舍五入到最接近的整数后，根据上述计算得出 4 和 7。

表 I 用于代码异味的代码质量度量指标及主观片段识别的低阈值和高阈值

代码异味	代码质量度量指标	度量指标阈值
复杂方法	圈复杂度	4–7
过长参数列表	参数数量	3–6
多面	方法缺乏内聚性（LCOM）	0.4–0.8
抽象	方法（LCOM）	

2) 第一阶段之后：我们根据第一阶段得出的阈值配置过滤机制，并在 Twitter 和 LinkedIn 等社交媒体平台上发布 TAGMAN 的安装链接，邀请标注人员参与。所有了解 Java 编程语言和至少具备基本面向对象概念的软件开发人员、软件工程专业学生和研究人员均可参加。我们在 2022—2023年12月至1月期间开放了六周的报名。共有 82 名标注人员参与了这一阶段。TAGMAN 展示了度量指标值处于低阈值和高阈值之间（含两端）的代码片段。我们将 TAGMAN 配置为每个样本获取两条标注，以提高标注的可靠性。

F. 数据集信息

第二阶段结束后，我们共收到 86 名标注人员针对 5，192 个样本提交的 10，267 条标注。表 II 列出了各代码异味类型对应的标注和样本数量。

数据集可用性：本文提供的数据集可在线获取 [32]。此外，用于准备代码片段的脚本所在代码仓库[2]以及代码标注应用程序[3] 即 TAGMAN 也可在线获取。

[3]https://github.com/SMART-Dal/Tagman

表 II 数据集元数据

数据集	代码异味	#标注	#样本
DACOS	复杂方法	4,349	2,197
	过长参数列表	3,221	1,634
	多面抽象	2,697	1,361
	总计	**10,267**	**5,192**
DACOSX	复杂方法	–	94,489
	过长参数列表	–	93,442
	多面抽象	–	19,674
	总计	**–**	**207,605**

III. 潜在研究应用

- **代码异味的检测与验证：** 鉴于代码异味具有主观性，传统的代码异味检测工具通过实现规则和启发式方法来识别代码异味，因此表现不佳。机器学习方法能否成功检测代码异味，取决于是否有大型人工标注数据集可供使用。本文介绍的数据集 DACOS 和 DACOSX 对现有数据集进行了补充，为现有数据集未涵盖的三种代码异味提供了大量主观代码片段。

- **将代码异味与软件工程各方面相关联：** 目前已有多种探索性和实证研究探讨代码异味的影响，包括缺陷预测 [33], 可维护性预测 [34],以及维护工作量 [35]。除现有研究方向外，基于所提供数据集训练或微调的工具还可用于有效建立代码异味与软件开发团队生产力之间的关系。

- **扩展 TAGMAN 以实现代码标注：** 我们开发的代码标注应用程序可用于创建该数据集，也可扩展到类似的代码标注任务，例如检测易受攻击的代码和筛选编写规范的标识符。

IV. 相关数据集

软件工程文献中可供使用的代码异味人工标注数据集为数不多。Palomba 等人 [36] 提供了 LANDFILL 数据集，其中包含五种代码异味的标注：发散式变更、霰弹式手术、平行继承、上帝类，以及特性依恋。他们为 243 个代码片段提供了标注。他们还开发了一个在线门户，供贡献者标注代码异味；然而，在本文撰写时（2023年1月），该门户无法访问。Madeyski等人 [37] 提出了 MLCQ——一个代码异味人工标注数据集。该数据集包含 4.7 千条标注，涵盖 4,770 个样本。该数据集考察了四种代码异味：上帝类、数据类、过长方法，以及特性依恋。上述两个数据集都没有考虑代码片段的主观性，因此，在用于训练机器学习分类器时，大多数代码片段可能无法提供任何新信息。此外，我们选择了任何现有代码异味数据集都未涵盖的代码异味，从而对现有数据集进行了补充。还有一些代码异味数据集，例如 QScored [23]。尽管 QScored 数据集规模庞大，但其中的样本并非人工标注，因此缺少对上下文的必要捕捉。

V. 有效性威胁

内部效度威胁关系到能否从我们的实验结果中得出结论。在人工标注的第二阶段，我们邀请了至少具备 Java 编程语言和面向对象概念基础知识的志愿者。我们通过社交媒体专业渠道（Twitter 和 LinkedIn）发布了邀请。鉴于这项工作的匿名性，我们没有任何机制来验证参与者是否具备足够的知识来完成标注。不过，我们邀请所有主要参与者（至少完成了 50 项标注）作为数据集贡献者；我们认为，这项措施能够激励标注人员尽其所能完成标注。此外，我们将 TAGMAN 配置为每个样本获取两份标注，以降低出现随机标注的可能性。

外部效度威胁关系到能否将我们的结果推广应用。所提出的数据集包含以 Java 编写的代码片段。不过，我们的代码标注工具具有通用性，可用于标注任何编程语言的代码片段。此外，用于生成各个代码片段的脚本可以自定义，以使用其他外部工具将代码拆分为方法和类。此外，标注流程中用于依据某项度量指标的低、高阈值筛选代码片段的阈值均可配置。

VI. 局限性、结论与未来工作

我们提供了 DACOS，这是一个经过人工标注的代码异味数据集，包含针对 5,267 个主观代码片段的 5,192 条标注。我们还提供了一个大型 DACOSX 数据集，其中包含 DACOS 中已有内容之外的确定无异味和确定有异味代码片段。本文介绍了 TAGMAN，这是一款可在类似上下文中重复使用的代码标注应用程序。

所提出的数据集涵盖三种代码异味。我们在数据集中选择的代码异味种类较少，因为与每种异味只有少量标注相比，为一种异味提供更多标注更为可取。此外，我们选择的异味类型并未包含在现有代码异味数据集中。我们将 TAGMAN 配置为每个样本获取两条标注。尽管这提高了数据集的可靠性，但有人可能会认为，这可能导致标注彼此矛盾。我们可以通过将每个样本的标注数量增加到三条来缓解这一局限；未来版本的数据集中将纳入这一机制。此外，我们还希望在未来扩大数据集的范围，包括编程语言、样本数量以及支持的异味种类。

参考文献

[1] M. Fowler，《重构：改进现有程序的设计》，第1版。Addison-Wesley Professional，1999。[2] M. Perepletchikov 和 C. Ryan，“一项受控实验：评估耦合对面向服务软件可维护性的影响”，IEEE 软件工程汇刊，第37卷，第4期，第449–465页，2011年8月。[3] G. Bavota、A. Qusef、R. Oliveto、A. De Lucia 和 D. Binkley，“单元测试异味分布及其对软件维护影响的实证分析”，载于 IEEE国际软件维护会议，ICSM，萨莱诺大学，意大利萨莱诺。IEEE，2012年12月，第56–65页。[4] D. I. Sjøberg、A. Yamashita、B. C. Anda、A. Mockus 和 T. Dyb˚a，“量化代码异味对维护工作量的影响”，IEEE软件工程汇刊，第39卷，1156，2013年8月，第1144页。[5] Z. Soh、A. Yamashita、F. Khomh 和 Y.-G. Gu´eh´eneuc，“代码异味是否会影响不同维护编程活动的工作量？”，载于2016年IEEE第23届软件分析、演化与再工程国际会议（SANER）。IEEE，2016年，第393–402页。[6] A. Ikama、V. Du、P. Belias、B. A. Muse、F. Khomh 和 M. Hamdaqa，“重新审视反模式对易故障性的影响：一项差异化复制研究”，载于2022年IEEE第22届源代码分析与操作国际工作会议（SCAM），2022年，第56–65页。[7] F. Khomh、M. Di Penta、Y.-G. Gu´eh´eneuc 和 G. Antoniol，“反模式对类变更和易故障性影响的探索性研究”，《实证软件工程》，第17卷，第3期，第243–275页，2012年6月。[8] T. Hall、M. Zhang、D. Bowes 和 Y. Sun，“某些代码异味对故障有显著但较小的影响”，ACM软件工程与方法学汇刊（TOSEM），第23卷，第4期，第33–39页，2014年9月。[9] F. Jaafar、Y.-G. Gu´eh´eneuc、S. Hamel 和 F. Khomh，“挖掘反模式依赖关系与易故障性之间的关系”，载于《逆向工程工作会议》论文集，WCRE，蒙特利尔理工学院，加拿大蒙特利尔。IEEE，2013年，第351–360页。[10] T. Sharma 和 D. Spinellis，“软件异味综述”，《系统与软件杂志》，第138卷，第158–173页，2018年。[11] U. Azadi、F. A. Fontana 和 M. Zanoni，“海报：通过 wekanose 进行基于机器学习的代码异味检测”，载于2018年IEEE/ACM第40届软件工程国际会议：伴随会议（ICSE- Companion），2018年，第288–289页。[12] F. Pecorelli、F. Palomba、D. Di Nucci 和 A. De Lucia，“比较基于启发式和机器学习的度量指标代码异味检测方法”，载于2019年IEEE/ACM第27届程序理解国际会议（ICPC），2019年，第93–104页。[13] D. Di Nucci、F. Palomba、D. A. Tamburri、A. Serebrenik 和 A. De Lu- cia，“使用机器学习技术检测代码异味：我们是否已取得进展？”，载于2018年IEEE第25届软件分析、演化与再工程国际会议（SANER），2018年，第612–621页。[14] A. Imran，“开源Java软件的设计异味检测与分析”，载于2019年IEEE国际软件维护与演化会议（ICSME），2019年，第644–648页。[15] T. Sharma、V. Efstathiou、P. Louridas 和 D. Spinellis，“通过深度直接学习和迁移学习进行代码异味检测”，《系统与软件杂志》，第176卷，文章编号110936，2021年。[在线]。可访问：https://www.sciencedirect.com/science/article/pii/S0164121221000339 [16] T. Sharma、M. Kechagia、S. Georgiou、R. Tiwari、I. Vats、H. Moazen 和 F. Sarro，“源代码分析机器学习技术综述”，2022 [ ] 2110 09610 。在线。可访问：https://arxiv.org/abs/。[17] F. Palomba、D. D. Nucci、M. Tufano、G. Bavota、R. Oliveto、D. Poshyvanyk 和 A. D. Lucia，“LANDFILL：一个带有公开评估的代码异味开放数据集”，2015年5月。[在线]。可访问：https://doi.org/10 5281 6080422 。/zenodo。[18] G. Suryanarayana、G. Samarthyam 和 T. Sharma，《软件设计异味重构：管理技术债务》，第1版。Morgan Kaufmann，2014年。[19] T. Sharma、M. Fragkoulis 和 D. Spinellis，“你的配置代码是否存在异味？”，载于《第13届国际软件仓库挖掘会议论文集》，MSR ’16 系列。美国纽约州纽约市：计算机协会，2016年，第189–200页。[在线]。获取地址：https://doi.org/10 1145 2901739 2901761 ./。[20] ——，“纸牌屋：开源 C# 代码仓库中的代码异味”，载于 2017 年 ACM/IEEE 国际实证软件工程与度量研讨会（ESEM），2017年，第424–429页。[21] T. Sharma，“searchgithubrepo——搜索 GitHub 代码仓库”，https:// github.com/tushartushar/search-repo，2022年。[22] “GitHub GraphQL API 文档。”[在线]。获取地址：https: // docs.github.com/en/graphql [23] T. Sharma 和 M. Kessentini，“QScored：大型代码异味与质量度量指标数据集”，载于2021年 IEEE/ACM 第18届国际软件仓库挖掘会议（MSR），2021年，第590–594页。[24] T. Sharma，“CODESPLITJAVA”，2019年2月，https://github.com/tushartushar/CodeSplitJava。[在线]。获取地址：https://doi.org/10 5281 2566865 . /zenodo。[25] T. Sharma、P. Mishra 和 R. Tiwari，“Designite——软件设计质量评估工具”，载于2016年 IEEE/ACM 第1届将架构设计思维融入开发者日常活动国际研讨会（BRIDGE），2016年，第1–4页。[26] W. Oizumi、L. Sousa、A. Oliveira、L. Carvalho、A. Garcia、T. Colanzi 和 R. Oliveira，“重构后的类中退化症状的密度与多样性：一项多案例研究”，载于2019年 IEEE 第30届国际软件可靠性工程研讨会（ISSRE）。IEEE，2019年，第346–357页。[27] T. Sharma、P. Singh 和 D. Spinellis，“设计异味与架构异味之间关系的实证调查”，《实证软件工程》，第25卷，第5期，第4020–4068页，2020年。[28] A. Eposhi、W. Oizumi、A. Garcia、L. Sousa、R. Oliveira 和 A. Oliveira，“通过重构消除设计问题：我们关注的是正确的症状吗？”，载于2019年 IEEE/ACM 第27届程序理解国际会议（ICPC），2019年，第148–153页。[29] A. Uchˆoa、C. Barbosa、W. Oizumi、P. Blenilio、R. Lima、A. Garcia 和 C. Bezerra，“现代代码审查如何影响软件设计退化？一项深入的实证研究”，载于2020年 IEEE 国际软件维护与演化会议（ICSME），2020年，第511–522页。[30] M. Alenezi 和 M. Zarour，“使用 Designite 工具对软件演化过程中坏味道的实证研究”，《i-Manager 软件工程期刊》，第12卷，第4期，第12页，2018年。[31] T. McCabe，“一种复杂度度量”，《IEEE软件工程汇刊》，第SE-2卷，第4期，第308–320页，1976年。[32] H. Nandani、M. Saad 和 T. Sharma，“DACOS——数据集”，2023年1月。[ ] 10 5281 7570428 在线。获取地址：https://doi.org/ . /zenodo。[33] F. Palomba、M. Zanoni、F. A. Fontana、A. De Lucia 和 R. Oliveto，“闻起来像青春精神：利用代码异味强度提升缺陷预测性能”，载于2016年 IEEE 国际软件维护与演化会议（ICSME），2016年，第244–255页。[34] A. Yamashita 和 L. Moonen，“代码异味是否反映了重要的可维护性方面？”，载于2012年 IEEE 第28届国际软件维护会议（ICSM），2012年，第306–315页。[35] D. I. K. Sjøberg、A. Yamashita、B. C. D. Anda、A. Mockus 和 T. Dyb˚a，“量化代码异味对维护工作量的影响”，《IEEE软件工程汇刊》，第39卷，1156，2013 8，1144 第 pp. – 。[36] F. Palomba、D. Di Nucci、M. Tufano、G. Bavota、R. Oliveto、D. Poshy- vanyk 和 A. De Lucia，“LANDFILL：带有公开评估的代码异味开放数据集”，载于2015年 IEEE/ACM 第12届软件仓库挖掘工作会议，2015年，第482–485页。[37] L. Madeyski 和 T. Lewowski，“MLCQ：行业相关代码异味数据集”，载于《软件工程评估与测量会议论文集》，EASE ’20 系列。美国纽约州纽约市：计算机协会，2020年，第342–347页。[在线]。获取地址：https://doi.org/10.1145/3383219.3383264