缺陷正在形成：我们离它有多近？一种时间敏感型预测方法

Mikel Robredo*、Matteo Esposito*、Fabio Palomba†、Rafael Pe˜naloza‡、Valentina Lenarduzzi§*,
*奥卢大学 — †萨莱诺大学 — ‡米兰-比可卡大学 — §南丹麦大学{mikel.robredo, matteo.esposito}@oulu.fi;
fpalomba@unisa.it; rafael.penalozanyssen@unimib.it; lenarduzzi@imada.sdu.dk

arXiv:2601.01921v1 [cs.SE] 5 Jan 2026

摘要—背景。**缺陷预测一直是实证软件工程领域研究人员高度关注的课题。以往研究通过即时预测，成功实现了对即将出现的故障进行最准确的预测，并识别出故障发生前的特征和异常。随着软件系统不断演进，人们越来越需要能够在缺陷显现之前预测缺陷的时间敏感型方法。** 目标。**本研究旨在探索时间敏感型技术在缺陷预测中的有效性。此外，我们还希望探究缺陷发生前的早期指标。** 方法。**我们将训练多种时间敏感型预测技术，以预测软件项目未来的程序错误密度，并识别缺陷发生前的早期症状。** 预期结果。**我们预期的结果将转化为实证证据，用以证明我们的方法在早期估计缺陷倾向性方面的有效性。**

索引词—**时间序列分析；软件维护；实证软件工程；Transformer；缺陷预测**

I. 引言

在软件维护和演进过程中，开发者经常会修改源代码以引入新功能或修复缺陷 [1] 。然而，这些修改可能会无意间引入新的缺陷 [2], ，因此开发者必须仔细验证，以确保此类更改不会在代码中引入问题。这项验证工作通常直接在开发过程中进行（例如，运行测试用例），或通过代码审查进行 [3]。

一种有效的策略是通过缺陷预测，将检查和测试资源分配给源代码中更可能存在缺陷的部分 [4]。缺陷预测涉及构建统计模型，以预测软件制品的缺陷倾向性，主要依据与源代码或开发过程相关的信息 [5]。

过去十年间，缺陷预测问题受到了研究人员的广泛关注。他们尝试通过两种主要途径解决这一问题：（i）开展实证研究，以确定导致软件制品更容易出现程序错误的因素；（ii）提出创新的预测模型，以准确预测源代码的缺陷倾向性。近期的缺陷预测研究尝试通过多种基于机器学习和深度学习算法的方法来应对这一挑战 [6]，并采用有监督和无监督技术 [7]。这些近期研究利用未来的二元分类进行即时（JIT）缺陷预测。它们按时间顺序排列训练数据，因此基于过去的历史数据进行在线预测 [8]–[10]。此外，先前研究还探讨了适用于缺陷预测的潜在模型无关解释技术 [11], ，并探索了借助深度学习使软件缺陷预测更加透明、可解释的方法 [12]，[13]。我们的研究旨在沿着同一路径，考察程序错误发生之前出现的早期症状 或软件指标。

尽管这些研究显著提升了我们对缺陷预测的理解，但它们都侧重于对缺陷是否发生进行二元预测。与此同时，现有的时间敏感型预测模型在概率估计事件发生方面已展现出良好的前景，例如时间序列分析（TSA）技术 [14], 、贝叶斯技术 [15], ，以及基于Transformer架构的新型技术 [16]。因此，我们旨在探索现有时间敏感型预测方法在概率估计缺陷发生方面的有效性，从而利用预先估计的知识**支持**现有即时缺陷预测技术的结果。此外，我们还旨在研究采用所考察方法时，缺陷发生之前出现的早期指标。

因此，在这份注册报告中，我们设计了一项**探索性研究**，以探究时间敏感技术在缺陷倾向性估计中的有效性，并进一步考察哪些早期症状会先于缺陷的引入出现。由此，我们旨在为拟议的研究提出三项主要贡献：

- 通过探索时间敏感技术在缺陷倾向性估计中的有效性，为当前关于故障/缺陷预测的知识提供补充。
- 探索不同时间窗口对缺陷倾向性估计有效性的影响。
- 详细识别在缺陷引入之前及期间，缺陷预测中最具描述性的自变量所呈现的指标。

论文结构。 第二节介绍背景与相关工作，第三节描述实证研究设计。第四节指出一些可能影响有效性的因素，第五节总结本文研究的目标，并展望未来工作。

II. 背景与相关工作

本节概述理论背景，并讨论与我们的实证研究相关的先前研究。

A. 软件缺陷预测

软件缺陷预测是软件工程领域一个长期存在的研究方向，已有大量文献探讨如何识别易出错组件 [7]。传统方法在发布时估计缺陷倾向性，而即时预测则将重点转向提交级粒度 [17],[18]。在这些基础上，连续型缺陷预测框架 [19] 和基于异常的技术 [20], [21]推动了对软件演化的大规模实证分析。事实证明，产品度量（例如，麦凯布复杂度 [22],和 CK 套件 [23]）以及过程度量（例如，代码变更量 [24],和熵 [25]）都是可靠的缺陷倾向性预测变量 [26]–[28]。后续研究强调了提交级特征的重要性，例如开发者经验、提交规模和代码熵 [29], ，并表明规模较大或较复杂的提交更有可能引入缺陷。近期的机器学习与深度学习模型进一步提高了预测准确性和可解释性 [5], [13], [30], [31], ，但它们本质上仍是被动响应的，侧重于在缺陷被引入后进行检测，而不是理解缺陷何时开始出现。因此，Jiarpakdee 等人 [11]开展了一项大规模实证研究，探讨模型无关解释技术，以了解实例级解释技术对于解释软件组件为何以及随后如何被预测为存在缺陷的价值。在此基础上，Khanan 等人 [12]提出了 JITBot，这是一款可解释的即时缺陷预测聊天机器人，不仅能够预测软件组件的缺陷倾向性，还能解释导致程序错误发生的因素。

虽然传统的缺陷预测研究主要侧重于判断某次变更或某个文件是否存在缺陷，通常是在发布时或提交时进行判断，但**我们的研究**将视角转向对缺陷进行时间感知且演化性的理解。我们不再将程序错误视为二元结果，而是对其时间演变过程进行建模，旨在预测缺陷可能何时产生。时间视角使我们能够识别缺陷形成的早期症状，捕捉程序错误显现之前出现的迹象。因此，我们的方法提供了关于缺陷产生与演化的连续型概率视图，从而**克服**了当前最先进缺陷预测模型静态且事后分析的局限，并为主动式质量保证和预防性维护带来了显著优势。

B. 缺陷生命周期回溯检查

以项目的问题跟踪系统（ITS） [29]中报告的受影响版本（AV）为依据，对缺陷提交进行标注的益处已在先前研究中得到探讨。由于问题单中并未系统地列出所有受影响版本，Vandehei 等人 [32] 和 Falessi 等人 [33]定义了稳定比例（P）方法。P 方法根据缺陷在各受影响版本中的演变时间维度计算得出（见图2）。在他们提出的方法中，缺陷首先在问题跟踪系统（ITS）中记录问题单之前，于注入版本 （IV）被引入代码库。随后，项目的问题跟踪系统（ITS）中会创建缺陷报告，并可根据日期将其与缺陷的问题创建版本（OV）对应起来。因此，当缺陷修复提交被记录到项目的版本控制历史（C_i） [34]中时，缺陷的稳定生命周期便告结束。因此，对于每个缺陷，IV 之前的版本会被标记为未受影响，而从 IV 到修复版本（FV）的版本则会被标记为受影响版本（AV）。因此，若问题单未详细记录 IV，便可通过启发式方法，根据发现并修复缺陷所需版本数量的比例来估算缺陷的稳定生命周期，即 $FV-OV$与 $FV-IV$成比例。作者倾向于采用这种方法，先计算 P 值，从而能够为软件项目中检测到的每个缺陷标记所有受影响版本（AV）。

C. 时间感知分析

a) 时间序列分析方法： 时间序列分析为理解时间有序数据 [35]提供了统计学基础。这些技术揭示数据随时间变化的依赖关系，从而能够根据历史模式预测未来趋势。模型的选择取决于非平稳性、季节性和外部影响等关键时间特性 [35]。

在软件缺陷预测中，TSA技术已被用于捕捉软件质量的时间动态。Wu等人 [36] 率先开展了这一方向的研究，将自回归积分滑动平均模型、X12增强型自回归积分滑动平均模型和多项式回归应用于Debian缺陷数据进行比较。结果表明，X12增强型自回归积分滑动平均模型取得了最佳的预测准确度。在此基础上，Pati和Shukla [37] 应用了单变量自回归神经网络和混合自回归积分滑动平均—神经网络模型，结果证明混合方法能够产生最准确的预测。

在这些研究基础上，我们的研究通过纳入额外的自变量，拓展了TSA的应用。通过这一方法，我们旨在识别软件缺陷的早期时间指标，并检测软件开发过程中逐渐显现的潜在缺陷症状。

b) 贝叶斯方法：贝叶斯方法尤其为处理软件度量中的不确定性和因果依赖关系提供了原则性的方法。Okutan和Yıldız [38]使用贝叶斯网络揭示了代码度量与缺陷倾向性之间的概率关系，表明基于网络的推理能够提高可解释性和稳定性。近期，Kumar等人 [39] 采用结合特征排序和CK度量的贝叶斯信念网络，在预测易故障模块方面达到了77.9%的准确率。与决策树或随机森林等经典分类器相比，贝叶斯模型在不同数据集上能产生更稳定的结果，并可将定量代码度量与定性过程因素结合起来 [40]。在我们的研究场景中，缺陷预测是在提交和文件级别上进行的，贝叶斯推断提供了一种自然的扩展方式，能够捕捉过程度量（例如，代码变更量、

熵）与缺陷结果之间的概率依赖关系，同时对不确定性进行显式建模。

c) 基于 *Transformer* 的方法：与此同时，基于 Transformer 的架构近期展现出在软件缺陷预测方面的巨大潜力，能够从源代码中捕获语法和语义信息。Liu 和 Zhou [41]提出了 DP-TFusion，这是一种将抽象语法树（AST）序列与基于度量的特征融合的 Transformer 模型，显著提升了跨版本预测性能。同样，Han 等人 [42] 表明，Transformer 模型优于循环模型和传统机器学习方法，尤其是在大型且类别不平衡的数据集上。此外，Zhang 等人 [43] 提出了一种分层 Transformer，可联合建模词元级和行级上下文，从而在行粒度上实现更高的精确率。在我们的研究中，基于 Transformer 的表示能够编码代码变更的序列特性和结构特性，捕获前后语句之间的依赖关系。

III. 实证研究设计

本节介绍我们设计的实证研究，包括研究目标与研究问题、研究背景、数据收集和数据分析（见图 1）。我们根据 Wohlin 等人 [44]制定的指南设计实证研究。在这项研究中，我们考察基于 TSA [14], 贝叶斯推断 [15],和现代基于 Transformer 的架构 [16], [45]的缺陷预测模型的有效性。为便于阅读，本文将这些方法统称为模型 。同样，为使全文更易于阅读，我们将所有**产品**、**过程**及其他**质量**度量统称为软件度量（SM）。

为了便于复现本研究，我们将公开原始数据。

A. 目标与研究问题

本实证研究的**目标** 是分析时间敏感型缺陷倾向预测，旨在评估其支持和补充现有缺陷预测文献的能力，就其能否更早提供缺陷发生的概率估计，以及能否识别缺陷引入前出现的早期指标（症状）而言；从研究人员和实践者的角度，在开源软件项目的背景下进行考察。

基于这一目标，我们提出第一个研究问题（**RQ**$_1$）：

> **RQ**$_1$**：** 时间敏感型预测模型在多大程度上能够估计项目中缺陷密度随时间的变化？

先前的研究考察了缺陷预测模型的预测性能 [36], [37], , 结果表明，时间序列分析（TSA）方法能够有效预测未来的缺陷数量。软件工程（SE）领域也报告了相关预测应用取得的积极成果 [14]。

我们的研究问题聚焦于：缺陷预测模型能否近似预测项目代码库中缺陷随时间的注入情况。为此，我们采用启发式基线，对项目版本控制历史中的缺陷提交进行标注。我们参考了先前关于缺陷倾向性的研究，以确定每次提交对应的缺陷数量 [32]–[34], , 并将其作为尽力而为的真实值，用于评估预测结果的近似效果。我们使用 MAPE、平均绝对误差（MAE）和均方根误差（RMSE）等标准误差指标来评估这种近似效果 [14],, 并在一步超前和下一观测时域两种设置下评估预测性能。

除了短期预测之外，我们还进一步研究缺陷预测模型补充当前缺陷预测知识的长期潜力。具体而言，我们分析其在多个未来时间跨度上的有效性。因此，我们提出：

> **研究问题**$_2$。预测时距的长短如何影响缺陷预测的准确性和稳定性？

时间敏感型缺陷预测的一项关键挑战是进行多步预测，即预测未来多个时间步的情况。长期缺陷预测可以通过揭示代码库中缺陷持续发生的概率，补充缺陷预测。单步预测适用于短期行动，而即时缺陷预测在这方面已表现出良好效果；多步预测则能进一步提供系统距离未来缺陷发生还有多近的概率视图。先前的研究已展示了这一潜力：[47] 实现了最长提前三个月的准确预测，而 [14]则将预测时距延长至三年。这类前瞻性分析有助于提升开发人员的态势感知能力，并促进更积极主动的软件维护实践 [6], [48]。

因此，在 RQ$_2$,中，我们旨在生成长期预测，以估计当前代码变更可能如何影响未来的缺陷发生。为此，我们在两种互补的设置下，评估多个时间范围内的预测表现。对于每个时间范围，我们计算标准误差指标——平均绝对百分比误差、平均绝对误差和均方根误差——以量化预测准确性。最后，为评估长期预测效应的显著性，我们对以下假设进行统计检验：

$H_{1.01}$不同模型在不同时间范围内的预测有效性没有差异。
$H_{1.11}$ 不同模型在不同时间范围内的预测有效性存在显著差异。

此外，我们还希望探究表现最佳的模型和时间窗口在统计上是否优于其他模型和窗口。因此，我们进行事后分析，以检验以下假设：

$H_{1.02}$在预

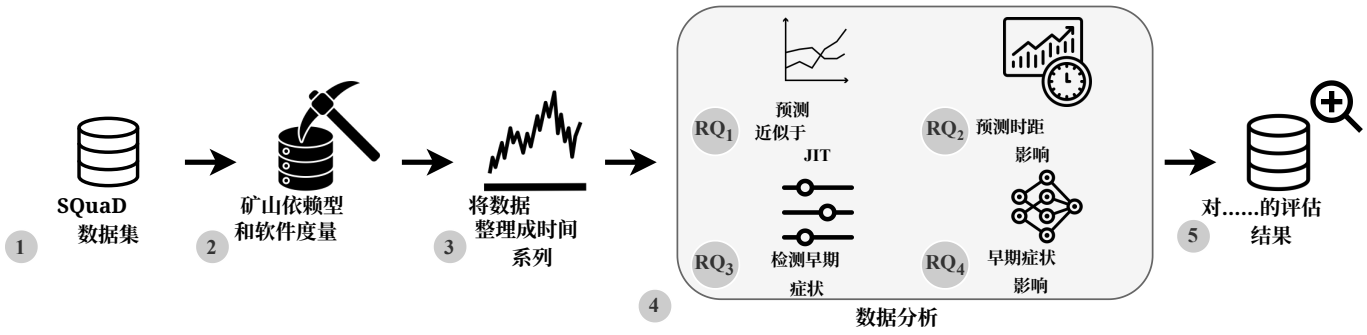


图 1. 研究设计概览（SQuaD：软件质量数据集）

测有效性方面，不同时间范围内任意两个模型之间都不存在显著的成对差异。$H_{1.12}$ 在不同时间范围内，至少有一对模型的预测有效性存在统计学显著差异。

长期预测的缺陷倾向性可能呈上升或下降趋势，但不会凭空发生。随着时间推移，不同的软件度量会呈现特定模式，这些模式或许能够解释缺陷倾向性概率的变化。因此，我们提出以下问题：

研究问题$_3$。 缺陷发生之前有哪些早期症状?

先前研究表明，软件缺陷源于软件演化过程中出现的多个相互关联因素 [30]。尽管多项研究从不同来源探讨了可能影响缺陷预测的因素 [26]⁋[28],，但当前最先进的研究主要聚焦于产品与过程软件度量，并将其视为关键决定因素 [28], [30]。近期研究进一步厘清了哪些过程软件度量最能表征缺陷发生 [33], [46]。

通过 RQ$_3$,，我们旨在根据预测结果，探究缺陷更可能被引入的条件，从而拓展现有的缺陷预测知识体系。具体而言，我们采用成熟的特征重要性技术来量化软件度量的相对重要性，包括随机森林（RF） [14],、极端梯度提升（XGB） [49], 、相关性分析[50], 和信息增益率（IGR） [34]。此外，我们将通过双管齐下的策略应用这些发现。首先，我们将采用专业人士偏好的预测窗口长度，包括**每周**、**每两周**和**每月**预测窗口长度 [14],，并将其应用于 TSA、贝叶斯和基于Transformer的模型。其次，我们将进行多步预测，评估模型在不断延长的预测时域上的预测性能。这将帮助从业者理解长期预测的前瞻水平会如何影响当前的软件维护决策。

最后，我们旨在评估信息量最大的 SM 的贡献，方法是考察移除它对缺陷预测模型性能的影响。因此，我们提出以下问题：

研究问题$_4$。缺陷发生前的早期症状是否有助于提升缺陷预测的有效性?

为了了解不同组件如何影响模型性能，研究人员通常会进行消融实验，即移除某个组件（如一项特征），并衡量其缺失对模型预测性能的影响 [51]。过去一年，这种方法的应用日益普遍 [52], [53]。因此，为了强化研究问题 RQ$_3$,的研究结果，我们开展了一项消融实验。具体而言，我们旨在通过使用研究问题 RQ$_2$中确定的最佳模型—时间窗口组合进行实验，评估最具信息量的症状，也就是最具信息量的软件度量（RQ$_3$）的贡献。为此，我们将针对软件度量的所有可能组合设计消融实验，以进一步证明这些度量的预测相关性是否体现在实际预测中，以及它们的缺失是否会导致预测准确度下降。

因此，我们提出以下**零**假设和**备择**假设：

$H_{4.01}$ 当最具信息量的症状被纳入预测变量时，模型性能没有差异。$H_{4.11}$ 当最具信息量的症状被纳入预测变量时，模型性能存在显著差异。

B. 研究背景

作为背景，我们计划考虑软件质量数据集（SQuaD） [54]中的项目列表。SQuaD 提供纵向提交级度量、根据成熟方法得出的缺陷标签，以及高质量的过程/产品度量，因此适用于时间敏感的缺陷预测研究。在其涵盖的 450 个开源软件（OSS）项目中，包括来自*Apache*软件基金会（ASF） [14], [34], 、*Mozilla* [55]等来源的代码仓库，以及*FFMpeg* 框架 [56],和 Linux 内核 [56]的代码仓库。此外，项目的筛选遵循

表 I 代码仓库挖掘标准。

标准	条件
已归档或已分叉	代码仓库既未归档，也不是分叉。
最近活动	代码仓库最近一次活动距今不到六个月。
贡献者	代码仓库至少有三位贡献者。
星标数	代码仓库至少有 50 个星标。

系统化挖掘标准（见表 I），以确保所挖掘项目的质量 [57]。

所选项目的更多详情请见此处[54]。

C. 变量

下文将介绍本研究所考察的因变量和自变量。

1）因变量：我们将从挖掘项目的代码提交中**检测到的缺陷数量**作为因变量。因此，该因变量的性质为连续型。该变量表示提交级别上当前存在的缺陷数量。

2）自变量：我们将已证明能够提升缺陷预测性能的**过程**度量 [28], [46]作为自变量。此外，我们还考虑了采用的挖掘工具 Understand [1] 收集的 111 项**产品**度量。遵循 SE 社区近期的做法 [50], ，我们依赖 Understand 收集的软件度量来训练模型。如前所述，为了提高可读性，我们将其称为软件度量。

D. 研究执行

本小节介绍执行计划，包括数据收集和分析策略。

数据提取：此过程由以下三个独立任务组成：

- 任务 *1*——采用 *Vandehei* 等人的回溯式缺陷标注方法收集缺陷 *[32]*（图2）：**– 步骤 1：** 根据提交消息 [58]，在每个项目的版本控制历史中识别缺陷修复提交（C_i）。**步骤 2：** 对于每个缺陷，我们从 SQuaD 数据集中收集相应的问题单及其注入版本（*IV*）（如有）。或者，我们使用 Vandehei 等人定义的稳定比例 P [32]，以启发式方法估算 IV。**– 步骤 3：** 利用已识别的 IV 和 C_i，为缺陷的受影响版本（*AV*）范围内的所有提交添加标注。因此，我们收集一份列表，记录每次提交时处于活动状态的缺陷。**– 注意：** 我们将提交用作时间锚点；不过，我们的分析并不关注单个代码变更，而是考察每次提交时的整个项目快照

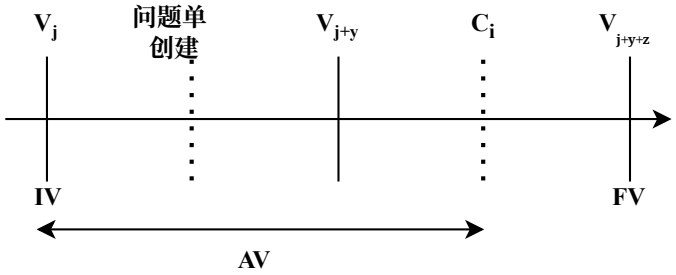


图 2. 缺陷生命周期示例：受影响版本（AV）、注入版本（IV）、缺陷修复版本（FV）、缺陷修复提交（C_i）。（*j, y, z*：提交观测点）

，并利用活动缺陷列表估算项目在该特定时点（即提交时间）的累计缺陷量。由此，我们可以连续地估算缺陷量。

- 任务 *2* - 过程度量挖掘：我们通过挖掘项目的完整变更历史，计算推荐用于提升缺陷预测准确性的 SM 软件度量指标 [28], [34], [46] 。
- 任务 *3* - 产品度量挖掘：我们在每个纳入研究的项目中使用 Understand 逐提交进行分析，并遵循现有方法 [54]，从而收集其完整变更历史中的 SM 软件度量值。

数据预处理：在本研究采用的模型中，时间序列分析模型需要经过时间序列化的观测数据才能进行训练 [35]。由于软件提交本质上并非周期性发生，我们将数据序列化为固定间隔，以便用于时间序列分析模型，从而为每个项目生成周期性时间序列 [14]。同样，遵循先前研究 [14], ，我们对非活动期（即没有提交的时期）产生的观测值采用线性数据插值。通过这一步预处理，我们旨在确保时间序列化的一致性与连续性，以满足运行时间序列分析模型的要求。

E. 数据分析

由于这是一份注册报告，我们尚未开展研究，因此无法事先 判断收集到的数据是否服从正态分布。因此，我们制定了一套数据分析方案，以兼顾正态和非正态条件。

a) 预测有效性（RQ_1 和 RQ_2）。：为了解决 RQ_1 和 RQ_2，我们以启发式缺陷推导方法的结果作为真实值，评估三个模型家族——时间序列分析（TSA）、贝叶斯推断和基于Transformer的架构——的预测有效性。对于每个模型–预测时域对，我们计算包括平均绝对百分比误差（MAPE）、平均绝对误差（MAE）和均方根误差（RMSE）在内的标准误差指标，以量化缺陷倾向性预测的准确性。我们采用滚动前向优化方法 [14]训练和测试模型。表II概述了本研究从各个已定义模型家族中选用的具体模型。此外，为了遵循

[1]https://docs.scitools.com/manuals/pdf/metrics.pdf

注册报告发表篇幅的限制，我们在共享的在线附录中进一步阐述所采用模型的定义、预测性能评估指标以及预期时间范围。[2]

我们首先使用安德森–达林检验（AD） [72]评估残差的分布。由于软件过程和预测数据通常会偏离正态性，我们主要采用威尔科克森符号秩检验（WT） [73]来评估不同模型和时间范围内预测性能的差异（$H_{1.01}$）。WT 是一种非参数检验，无需假设特定分布即可比较配对样本或独立样本，因此非常适合我们的数据情境。如果我们无法接受**零假设**，则使用不设对照组的邓恩检验进行事后比较（$H_{1.02}$）。邓恩检验是一种非参数事后检验方法，用于在基于秩的检验（如 WT）得出显著总体检验结果后识别具体的组间差异；它通过评估所有可能的组别组合，将检验扩展至多组两两比较，且无需设立对照组。

反之，如果 AD 检验使我们接受**零假设**，即数据服从正态分布，我们便采用参数检验来评估模型性能的差异。具体而言，我们使用单因素方差分析评估不同模型和时间范围内的预测准确度是否存在显著差异。方差分析使用F统计量检验均值相等的零假设，比较不同模型–时间范围组合的性能指标——平均绝对百分比误差、平均绝对误差和均方根误差。若方差分析显示差异具有统计显著性，我们将进一步采用 Tukey 诚实显著差异（**HSD**）事后检验，以确定哪些模型配对之间存在显著差异。

b) 信息量最大的症状（RQ_3）。： 为探究 RQ_3，我们考察哪些软件度量最显著地影响缺陷倾向预测。我们采用成熟的方法计算特征重要性，包括随机森林 (RF) [14]，极端梯度提升 (XGB) [49]，相关性分析 [50]，和信息增益率 (IGR) [34]。这些方法揭示了在不同模型和预测时域中，哪些软件度量最持续地助力准确预测。

c) 消融实验 *(RQ_4)*：对于 RQ_4，我们开展消融研究，以量化信息量最高的软件度量对模型的贡献。我们使用在 RQ_2 中确定的最佳模型 – 预测时域对，比较移除这些排名靠前的特征前后的预测性能。因此，我们使用 WT 检验 $H_{4.01}$，以检测模型准确率是否存在显著差异，因为该检验能够有效捕捉配对且非正态分布样本的差异。与 RQ_1 类似，对于正态分布的数据，我们使用配对t检验比较移除信息量最高的软件度量前后模型的预测性能。配对t检验用于评估两种配置下的性能指标（如平均绝对百分比误差、均方根误差）均值差异是否显著异于零，从而直接证明有多少预测准确性取决于已识别的关键软件度量。如果t检验显示存在显著变化，我们便将其解读为证据，表明被移除的软件度量对模型性能有重大贡献。

在所有统计检验中，我们采用 α =0.01 的显著性水平，以在第一类错误和第二类错误之间保持稳健的平衡 [74]。

F. 可重复性

为确保可重复性，我们发布原始数据和用于复现实验的代码，并将其打包在复现包中。

注入版本. 效度威胁

本节将按照实证软件工程研究中定义的类别，讨论本研究面临的主要有效性威胁 [44]。

构念效度。 尽管软件质量数据集提供了高质量的代码仓库，但各项目提交活动的不规律可能导致生成的时间序列中出现数据缺失。我们通过应用线性插值和数据插补来确保时间连续性，以缓解这一威胁。然而，这些技术得到的只是对实际值的近似；如果项目活动随时间变化更加稳定，结果可能会有所不同。此外，我们在所有项目中以一致的方式运用软件度量，以确保测量效度，并避免对所研究构念产生概念上的歧义。

内部效度。 我们将代码提交视为主要观测单位，因为故障会在这些节点被引入或检测到。我们依据先前研究报告的、与缺陷倾向性之间强烈的经验关联，选择了自变量——过程度量和产品度量。不过，我们无法排除其他未观测因素（例如团队实践、代码审查动态）可能影响结果。为缓解这一问题，我们开展消融分析（RQ_4），以量化信息量最高的软件度量的贡献，并验证它们与模型性能之间的因果相关性。

外部效度。 研究背景涵盖了使用多种广泛应用的编程语言（例如 Java、Python 或 C++）编写的成熟开源项目，涉及框架、实用工具和核心基础设施系统等不同领域。因此，研究结果可以推广到具有这些特征的多学科项目。不过，对于早期阶段、专有或活跃度较低的项目，研究结果可能并不适用，因为这些项目的开发模式和缺陷动态存在显著差异。

可靠性 有效性。 为确保可复现性，我们使用标准化工具和最先进的缺陷倾向性方法（例如 Understand，以及通过启发式缺陷推导方法进行缺陷倾向提交检测）自动提取所有软件度量和缺陷，并在我们的复现包中记录每个预处理步骤。所有脚本、配置和统计分析流程均已公开，确保其他研究人员能够独立复现并验证我们的结果。

[2]https://doi.org/10.5281/zenodo.17909376

表II 所采用预测模型概览。

模型类别	模型	描述
TSA	自回归积分滑动平均模型	通过差分和自回归建模捕捉趋势与自相关的单变量时间序列分析（TSA）模型 [14]。
	带外生变量的自回归积分滑动平均模型	在自回归积分滑动平均模型的基础上纳入自变量的多变量扩展模型 [59]。
	季节性自回归积分滑动平均模型	在自回归积分滑动平均模型的基础上，除了趋势之外，还对随机季节性模式进行显式建模 [14]。
	季节性带外生变量的自回归积分滑动平均模型	结合自变量和季节性模式的自回归积分滑动平均模型的季节性多变量扩展模型 [60]。
贝叶斯	BDLT	采用阻尼趋势以限制长期增长的结构时间序列模型 [61]。
	BETS	对水平、趋势和季节性进行近期加权的贝叶斯指数平滑 [62]。
	BDLM	支持时变回归系数的状态空间模型 [63]。
	BDGLM	BDLM 的扩展，支持对数据的非正态分布进行建模 [64]。
Transformer	TIMEGPT	在多种时间序列上预训练的基础模型 [65], [66]。
	LAG-LLAMA	用于单变量预测的仅解码器概率模型 [67]。
	CHRONOS	将实值时间序列分词，以利用 T5 语言模型架构 [68], [69]。
	MOIRAI	采用基于补丁的分词的概率掩码编码器模型 [70]。
	TimesFM	生成点预测的确定性仅解码器基础模型 [71]。

五、结论

本研究设计旨在实现两个互补目标。首先，提出一套全面的缺陷预测预测协议，整合不同的模型家族，包括时间序列分析、贝叶斯推断和基于 Transformer 的架构。通过这一协议，我们旨在评估提交活动中的时间模式和概率模式在多大程度上能够解释并预测故障发生。其次，本研究考察软件度量对缺陷倾向性的影响，识别缺陷引入之前出现的早期指标，并通过特征重要性和消融分析量化其影响。总体而言，这些目标为理解软件演化动态与软件度量之间的交互如何随时间塑造故障出现提供了结构化基础。

参考文献

[1] M. Lehman，《程序、生命周期与软件演化定律》，*IEEE Software*，第68卷，第9期，第1060–1076页，1980年。[2] S. Kim、E. J. Whitehead 和 Y. Zhang，《软件变更分类：干净还是有缺陷？》，*IEEE*软件工程汇刊，第34卷，第2期，第181–196页，2008年。[3] A. Bacchelli 和 C. Bird，《现代代码审查的期望、结果与挑战》，载于国际软件工程会议（*ICSE*），2013年，第712–721页。[4] T. Hall、S. Beecham、D. Bowes、D. Gray 和 S. Counsell，《软件工程中故障预测性能的系统性文献综述》，*IEEE*软件工程汇刊，第38卷，第1276–1304页，2012年。[5] L. Pascarella、F. Palomba 和 A. Bacchelli，《细粒度即时缺陷预测》，《系统与软件杂志》，第150卷，36，2019年，22页。– 。[6] Y. Zhao、K. Damevski 和 H. Chen，《即时软件缺陷预测的系统性综述》，*ACM*计算调查，第55卷，第10期，第1–35页，2023年。[7] N. Li、M. Shepperd 和 Y. Guo，《软件缺陷预测无监督学习技术的系统性综述》，《信息与软件技术》，2020年。[8] S. McIntosh 和 Y. Kamei，《诱发修复的变更是否是一个移动目标？即时缺陷预测的纵向案例研究》，载于第*40*届国际软件工程会议论文集，2018年，第560–560页。[9] S. Tabassum、L. L. Minku、D. Feng、G. G. Cabral 和 L. Song，《在线即时软件缺陷预测中的跨项目学习研究》，载于*ACM/IEEE*第*42*届国际软件工程会议论文集，2020年，第554–565页。

[10] M. Tan、L. Tan、S. Dara 和 C. Mayeux，《面向不平衡数据的在线缺陷预测》，载于《国际软件工程会议》，第2卷。IEEE，2015年，第99–108页。[11] J. Jiarpakdee、C. K. Tantithamthavorn、H. K. Dam 和 J. C. Grundy，《缺陷预测模型中模型无关技术的实证研究》，《IEEE软件工程汇刊》，第48卷，第2期，第166–185页，2022年。

[ ] 10 1109 2020 2982385 在线。可访问：[12] https://doi.org/ 。 /TSE. 。C. Khanan、W. Luewichana、K. Pruktharathikoon、J. Jiarpakdee、C. Tantithamthavorn、M. Choetkiertikul、C. Ragkhitwetsagul 和 T. Sunetnanta，《Jitbot：可解释的即时缺陷预测机器人》，载于第35届IEEE/ACM国际自动化软件工程会议（ASE 2020），澳大利亚墨尔本，2020年9月21日至25日。IEEE，2020年，第1336–1339页。 [在线]。可访问：https://doi.org/

10 1145 3324884 3415295 。 / 。T. Hoang、H. K. Dam、Y. [13] Kamei、D. Lo 和 N. Ubayashi，《DeepJIT：用于即时缺陷预测的端到端深度学习框架》，载于《第16届国际软件仓库挖掘会议论文集》（MSR 2019），2019年5月26日至27日，加拿大蒙特利尔，M. D. Storey、B. Adams 和 S. Haiduc 编。IEEE / ACM，2019年，第

34 45 [ ] 10 1109 2019 00016 – 页。在 [14] 线。可访问：https://doi.org/ 。 /MSR. 。M. Robredo、N. Saarimaki、D. Taibi、R. Penaloza 和 V. Lenarduzzi，《评估代码技术债务预测中的时变方法和季节性效应》，arXiv预印本 arXiv:2408.08095，2024年。[15] P. J. Harrison 和 C. F. Stevens，《贝叶斯预测》，《英国皇家统计学会会刊B辑：统计方法学》，第38卷，第3期，第205–228页，1976年。[16] M. Peixeiro，《使用基础模型进行时间序列预测》。Manning，2026年。 [在线]。可访问：https://books.google.fi/books?id=VWKOEQAAQBAJ [17] Y. Fan 等，《被szz错误标记的变更对即时缺陷预测的影响》，《IEEE软件工程汇刊》，2019年。[18] A. Mockus 和 D. M. Weiss，《预测软件变更风险》，《贝尔实验室技术期刊》，第5卷，2000年。[19] L. Madeyski 和 M. Kawalerowicz，《持续缺陷预测：构想及相关数据集》，载于《国际软件仓库挖掘会议》，2017年，第515–518页。[20] K. N. Neela 等，《将软件缺陷建模为异常：一项关于Promise仓库的案例研究》。《JSW》，2017年。[21] P. Afric 等，《Repd：基于异常检测的源代码缺陷预测》，《系统与软件杂志》，第168卷，2020年。[22] T. J. McCabe，《一种复杂度度量》，《IEEE软件工程汇刊》，第4期，第308–320页，1976年。[23] S. R. Chidamber 和 C. F. Kemerer，《面向对象设计的度量套件》，《IEEE软件工程汇刊》，第20卷，1994年。[24] J. Liu 等，《代码变更量：考虑工作量的即时缺陷预测中被忽视的度量》，载于《实证软件工程与度量国际研讨会》，2017年，第11–19页。[25] A. E. Hassan，《利用代码变更复杂度预测故障》，载于《国际软件工程会议》，2009年，第78–88页。[26] V. Basili 等，《面向对象设计度量作为质量指标的验证》，《IEEE软件工程汇刊》，第22卷，1996年。[27] T. Graves 等，《利用软件变更历史预测故障发生率》，《IEEE软件工程汇刊》，第26卷，2000年。

[28] L. Pascarella、F. Palomba 和 A. Bacchelli，《关于方法级缺陷预测的表现：一项负面结果》，《系统与软件杂志》，第161卷，2020年。[29] J. ´Sliwerski 等，《变更何时会引入修复？》，载于《软件仓库挖掘国际研讨会》，2005年。[30] Y. Kamei 等，《即时质量保证的大规模实证研究》，《软件工程汇刊》，2012年。[31] F. Lomio、S. Moreschini 和 V. Lenarduzzi，《基于SonarQube规则、产品度量和过程度量的机器学习与深度学习故障预测分析》，《实证软件工程》，第27卷，第7期，第189页，2022年。[32] B. Vandehei、D. A. D. Costa 和 D. Falessi，《利用缺陷生命周期标记受影响版本和缺陷类》，《 AC M软件工程与方法学汇刊》（TOSEM），第30卷，第2期，第1–35页，2021年。[33] D. Falessi、A. Ahluwalia 和 M. D. Penta，《休眠缺陷对缺陷预测的影响：对19个Apache项目的研究》，《软件工程与方法学汇刊》，第31卷，26，2021年1，1第。–。[34] M. Esposito 和 D. Falessi，《揭示隐藏风险：预测未修改方法中的缺陷倾向的重要性》，载于《2023年IEEE第23届源代码分析与处理国际工作会议》（SCAM）。IEEE，2023年，第277–282页。[35] P. J. Brockwell、R. A. Davis 和 M. V. Calder，《时间序列与预测导论》。Springer，2002年，第2卷。[36] W. Wu、W. Zhang、Y. Yang 和 Q. Wang，《用于缺陷数量预测的时间序列分析》，载于《第2届软件工程与数据挖掘国际会议》。IEEE，2010年，第589–596页。[37] J. Pati 和 K. Shukla，《使用自回归神经网络对Debian缺陷数据进行时间序列预测》，载于《2013年第4届计算机与通信技术国际会议》（ICCCT）。IEEE，2013年，第110–115页。[38] A. Okutan 和 O. Yıldız，《使用贝叶斯网络进行软件缺陷预测》，《实证软件工程》，第19卷，第1期，第154–181页，2014年。[39] S. Kumar、P. Singh 和 R. Kaur，《基于贝叶斯信念网络和CK度量特征排序的软件故障预测》，《智能与模糊系统杂志》，第49卷，第2期，第2351–2364页，2025年。[40] N. Sharma、G. Singh 和 P. Grover，《基于贝叶斯网络的软件可靠性与缺陷预测方法》，《数学》，第11卷，第11期，第2524页，2023年。[41] W. Liu 和 T. Zhou，《 DP-TFusion：一种基于Transformer的跨版本软件缺陷预测模型》，《神经计算与应用》，2025年，即将出版。[42] Y. Han、D. Kim 和 H. Park，《基于Transformer的混合软件缺陷预测模型》，《信息系统与电子商务管理杂志》，第23卷，第1期，第83–97页，2025年。[43] R. Zhang、L. Chen 和 Z. Wu，《用于细粒度软件缺陷预测的层次化Transformer》，arXiv预印本 arXiv:2312 11889，2023年。[44] C. Wohlin、P. Runeson、M. H¨ost、M. C. Ohlsson、B. Regnell 和 A. Wessl´en，《软件工程实验》，第二版。Springer，2024年。 [在线]。可访问：https://doi.org/10.1007/978-3-662 -69306-3[45] Y. Liu、H. Zhang、C. Li、X. Huang、J. Wang 和 M. Long，《Timer：生成式预训练Transformer是大型时间序列模型》，arXiv预印本 arXiv:2402.02368，2024年。[46] D. Falessi、S. M. Laureani、J. C。arka、M. Esposito 和 D. A. d. Costa，《通过即时预测提升方法和类的缺陷性预测》，《实证软件工程》，第28卷，第2期，第37页，2023年。[47] H. Jahanshahi、M. Cevik 和 A. Bas¸ar，《预测软件仓库中已报告缺陷的数量》，载于《加拿大人工智能会议》。Springer，2020年，第309–320页。[48] D. Di Nucci 等，《以开发者为中心的缺陷预测模型》，《IEEE软件工程汇刊》，第44卷，第1期，第5–24页，2017年。[49] M. Robredo、M. Esposito、F. Palomba、R. Pe˜naloza 和 V. Lenarduzzi，《你当时在想什么？一项由大语言模型驱动的开源项目重构动机大规模研究》，arXiv预印本 arXiv:2509 07763，2025年。[50] A. Bakhtin、M. Esposito、V. Lenarduzzi 和 D. Taibi，《网络中心性：微服务架构的新视角》，载于《2025年IEEE第22届国际软件架构会议》（ICSA）。IEEE，2025年，第72–83页。[51] S. Sheikholeslami，《面向机器学习的消融编程》，2019年。[52] Y. Wang、Z. Zhu、Q. Fu、Y. Ma 和 P. He，“Mrca：基于多模态数据的微服务指标级根因分析”，载于《第39届IEEE/ACM自动化软件工程国际会议论文集》，2024年，第1057–1068页。[53] Z. Cai、H. Wu、X. Jiang、X. Li 和 R. Buyya，“基于深度学习和反馈控制的云原生微服务容器自动伸缩”，《IEEE服务计算汇刊》，2025年。[54] M. Robredo、M. Esposito、D. Taibi、R. Pe˜naloza 和 V. Lenarduzzi，“Squad：软件质量数据集——数据集”，2025年11月。 [在线]。可访问：https://doi.org/ 10 5281 17566691 。/zenodo。A. [55] ´Lamkanfi、J. Perez 和 S. Demeyer，“Eclipse 和 Mozilla 缺陷跟踪数据集：用于挖掘缺陷信息的真实数据集”，载于《2013年第10届软件仓库挖掘工作会议（MSR）》。IEEE，2013年，第203–206页。[56] M. Liang、W. Charoenwet 和 P. Thongtanunam，“精选的基于电子邮件的代码审查数据集”，载于《第21届软件仓库挖掘国际会议论文集》，2024年，第294–298页。[57] E. Kalliamvakou、G. Gousios、K. Blincoe、L. Singer、D. M. German 和 D. Damian，“挖掘GitHub的前景与风险”，载于《第11届软件仓库挖掘工作会议论文集》，2014年，第92–101页。[58] M. Robredo Manero、M. Esposito、F. Palomba、R. Pe˜naloza 和 V. Lenarduzzi，“分析重构的连锁影响：一份注册报告”，本注册报告已被ICSME接收，2024年。[59] M. Mathioudaki、D. Tsoukalas、M. Siavvas 和 D. Kehagias，“比较用于技术债务预测的单变量和多变量时间序列模型”，载于《计算科学及其应用》，2022年，第62–78页。[60] G. E. Box、S. C. Hillmer 和 G. C. Tiao，“季节性时间序列的分析与建模”，载于《经济时间序列的季节性分析》。NBER，1978年，第309–344页。[61] A. C. Harvey，《波动率和重尾动态模型：及其在金融和经济时间序列中的应用》。剑桥大学出版社，2013年，第52卷。[62] R. J. Hyndman、M. Akram 和 B. C. Archibald，“指数平滑模型的可容许参数空间”，《统计数学研究所年刊》，第60卷，第2期，第407–426页，2008年。[63] J. Nakajima 和 M. West，“潜在阈值动态模型的贝叶斯分析”，《商业与经济统计杂志》，第31卷，第2期，第151–164页，2013年。[64] M. West、P. J. Harrison 和 H. S. Migon，“动态广义线性模型与贝叶斯预测”，《美国统计学会杂志》，第80卷，第389期，第73–83页，1985年。[65] A. Garza、C. Challu 和 M. Mergenthaler-Canseco，“Timegpt-1”，arXiv预印本 arXiv:2310.03589，2023年。[66] M. Peixeiro，《Python中的时间序列预测》。西蒙与舒斯特出版社，2022年。[67] K. Rasul、A. Ashok、A. R. Williams、A. Khorasani、G. Adamopoulos、R. Bhagwatkar、M. Bilo˜s、H. Ghonia、N. Hassen、A. Schneider 等，“Lag-llama：迈向时间序列预测基础模型”，载于《R0-FoMo：大型基础模型中少样本和零样本学习的鲁棒性》，2023年。[68] A. F. Ansari、L. Stella、C. Turkmen、X. Zhang、P. Mercado、H. Shen、O. Shchur、S. S. Rangapuram、S. P. Arango、S. Kapoor 等，“Chronos：学习时间序列语言”，arXiv预印本 arXiv:2403 07815，2024年。。[69] A. Roberts 和 C. Raffel，“探索T5的迁移学习：文本到文本迁移Transformer”，Google AI博客，2020年。[70] G. Woo、C. Liu、A. Kumar、C. Xiong、S. Savarese 和 D. Sahoo，“通用时间序列预测Transformer的统一训练”，2024年。[71] A. Das、W. Kong、R. Sen 和 Y. Zhou，“用于时间序列预测的仅解码器基础模型”，载于《第四十一届国际机器学习会议》，2024年。[72] T. W. Anderson 和 D. A. Darling，“基于随机过程的某些‘拟合优度’判据的渐近理论”，《数学统计年刊》，第23卷，第2期，第193–212页，1952年。 [ ] 10 1214 1177729437 在线。可访问：https://doi.org/。/aoms/。F. Wilcoxon，“基于秩方法的个体比较”，《生物统计学》，第1卷，第6期，第80页，1945年。 [在线]。可访问：https://app.dimensions.ai/details/publication/pub.1102728208 [74] P. Mishra、C. M. Pandey、U. Singh、A. Gupta、C. Sahu 和 A. Keshri，“统计数据的描述性统计与正态性检验”，《心脏麻醉年刊》，第22卷，第1期，第67–72页，2019年。