计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.250200052

工业物联网环境下软件缺陷预测技术的发展与应用综述

邓 涛[1] 邓 烨[2]
1 广西大学计算机与电子信息学院 南宁 530004
2 梧州学院电子与信息工程学院 广西 梧州 543003
(2313393012@st.gxu.edu.cn)

摘 要 在工业物联网(Industrial Internet of Things,IIoT)环境中,海量的软件代码数据的生成迫切需要通过先进的软件缺陷预测(Software Defect Prediction,SDP)技术进行有效分析。这些技术不仅能够迅速定位异常情况,还可以全面调查潜在问题,因为即使是微小的偏差也可能导致项目代码的崩溃。文中系统综述了2018—2025年间发表的61篇相关文献,突出展示了IIoT中SDP所面临的主要挑战和最新进展。从多个视角深入探讨了SDP的相关技术,包括统计方法、机器学习技术和模型导向的方法等。未来的研究应优先关注复杂异构环境中缺陷模式的动态变化,解决数据稀缺和标注成本高昂的问题,同时平衡实时性与资源限制之间的矛盾。此外,需要增强模型的可解释性和用户的认知理解,以提升系统的可理解性和操作的鲁棒性。还对IIoT中相关的现有数据集进行了系统分析,为该关键领域的进一步研究奠定了坚实基础。

关键词:工业物联网;软件缺陷预测;模型导向
中图分类号 TP311

Review of Development and Application of Software Defect Prediction Techniques in Industrial Internet of Things Environment

DENG Tao[1] and DENG Ye[2]
1 School of Computer,Electronics and Information,Guangxi University,Nanning 530004,China
2 School of Electronics and Information Engineering,Wuzhou University,Wuzhou,Guangxi 543003,China

Abstract In the context of IIoT,the generation of vast amounts of software code data necessitates effective analysis through advanced SDP techniques. These techniques not only enable the rapid identification of anomalies but also facilitate comprehensive investigations into potential issues,as even minor deviations can lead to significant code failures. This paper systematically reviews over 61 relevant articles published between 2018 and 2025,highlighting the primary challenges and recent advancements in SDP within IIoT. Various perspectives on SDP technologies are explored,including statistical methods,machine learning approaches,and model-oriented techniques. Future research should prioritize the dynamics of defect patterns in complex heterogeneous environments,address the challenges of data scarcity and high labeling costs,and balance the trade-off between real-time processing and resource constraints. Additionally,the interpretability of models and user cognitive understanding must be enhanced to improve system comprehensibility and operational robustness. A comprehensive analysis of existing datasets related to IIoT is also presented,laying a solid foundation for further research in this critical area.

Keywords Industrial Internet of Things,Software defect prediction,Model-oriented

1 引言

在工业4.0时代的快速发展背景下,工业物联网的智能化进程已成为其不可或缺的组成部分,其中嵌入式软件是边缘设备发展的核心要素。软件测试被视为确保嵌入式软件质量的关键环节,软件缺陷的识别与消除则被视为测试阶段的重要任务[1]。高质量与可靠性是软件产品的基本特性,但故障的出现会降低整体质量,从而对产品的可靠性产生负面影响,最终影响客户的满意度[2]。在软件开发的各个阶段,包括需求收集、设计、编码和维护,均存在潜在的软件缺陷风险[3]。

软件缺陷预测技术通过提供高效的机制,显著提升开发者和测试人员在软件开发周期中对缺陷的理解与控制能力。SDP方法运用统计分析模型、机器学习、深度学习及多种数据挖掘技术,旨在实现对潜在缺陷的早期识别和定量风险评估[4]。早期研究依托统计学的严谨性,以缺陷密度与故障率为主要观测指标[5],开展缺陷分析与预测工作。随着机器学习技术的兴起,基于软件度量和特征提取构建的预测方法逐渐成为SDP领域的主流。Devale等[6]提出的基于机器学习和即时策略的有效软件缺陷跟踪技术,为软件开发中的缺陷追踪提供了新的视角。Zheng等[7]针对即时软件缺陷预测模型的可解释性进行了探索,强调可解释性与可靠性是当前研究中亟待解决的主要问题。近年来,深度学习方法也展现出了强大的潜力并被广泛关注[8-9]。尽管SDP领域取得了显著的研究成果,但如何确保数据的质量与适用性,以及如何进行细致的模型训练和严格的验证流程,仍然是提升预测准确性的关键[10]。

为奠定后续深入探讨本文核心议题相关重要学术成果的基础,现将亟需关注的关键研究问题阐述如下:

1)现代方法相比于传统的SDP算法,在工业物联网中取

通信作者:邓烨(ShaYeiii@outlook.com)

得了哪些进展?

2)对于软件缺陷预测中应用的技术,如何在效率和精确度这两个方面进行评估?

3)现有的软件缺陷预测方法在处理缺陷时存在哪些局限性?

对本文与其他相关研究进行了详细的比较分析,如表 1 所列。与以往的研究不同,本实验不仅对现有的统计学方法、机器学习技术以及模型导向策略在 SDP 中的应用进行了全面的综合分析,还进一步深入评估了这些方法在不同应用场景下的表现、各自的局限性以及所使用的数据集特征。这一研究不仅为理解各类方法的有效性提供了更为系统的视角,同时也为未来的研究指明了改进与优化的方向。

本综述的主要目标是明确比较现有的 SDP 方法,突出它们的优势和局限性,并考察在推动该研究领域发展过程中的潜在挑战。本研究的主要贡献总结如下:

1)对 2018 年至 2025 年间发表的 61 篇聚焦于 IIoT 中的 SDP 研究文章进行了全面回顾,系统总结了该领域的最新进展。

2)本文对现有的 SDP 方法进行了深入分析,比较了统计方法、机器学习算法和深度学习技术,并评估了它们在不同场景中的适用性。同时,识别了关键的研究空白,并提出了未来的研究方向,为推动学术界和工业界的发展提供了重要见解。

3)此外,将深度学习方法分为 4 种主要类型:基于卷积神经网络、基于循环神经网络、基于对抗神经网络以及基于大语言模型的方法。分析涵盖了用于 SDP 的公开可用数据集,以确保研究的可重复性和结果的可靠性。

本文第 2 章对本研究采用的研究方法进行了概述;第 3 章概述了本文对所选的文章的分析结果;第 4 章系统回顾和分类了公开可用的 SDP 数据集;第 5 章基于当前研究结果探讨了 SDP 在 IIoT 领域的未来方向和挑战;最后总结全文。

2 研究方法

本章对选定文献进行了全面评述。在文献筛选阶段,本文严格界定了包容性和排除标准,确保了研究主题的精准定位与目标的集中性。基于既定标准,经过细致的质量评估,本文挑选出了符合标准的研究论文。

2.1 文章选择

本文聚焦于 IIoT 中的 SDP 问题,将软件缺陷预测分析方法分为 3 大类:统计方法、机器学习方法和深度学习方法。对这 3 类方法进行了全面的比较分析,并深入探讨了潜在的未来挑战。

表 1 本文与其他研究的比较概览

Table 1 Overview of this article compared to other researches

	维度						
	研究类型	传统方法	机器学习	模型导向	应用场景	局限性	数据集
Our study	Slr	√	√	√	√	√	√
Wu 等[11]	Survey	×	√	×	×	×	√
Li 等[12]	Survey	×	√	×	×	×	×
Liu 等[13]	Slr	√	√	×	√	√	√
Cai 等[14]	Survey	√	√	×	√	√	×
Batool 等[15]	Slr	√	√	×	×	×	×
Zain 等[16]	Slr	√	√	×	×	√	×
Pandey 等[17]	Slr	√	√	×	×	√	×
Rathore 等[18]	Survey	×	√	×	×	√	×

注:"Slr"为 Systematic Literature Review 的简写。

2.1.1 纳入标准

为了深入探讨当前研究主题的关键问题,特意选择了"软件缺陷预测""软件缺陷"和"工业物联网"作为主要搜索关键词。在四大主要学术搜索平台——"Google Scholar""ScienceDirect""ACM Digital Library""Web of Science"以及中国知网上进行了广泛的文献搜索,涵盖了 2018 年至 2025 年的相关论文。

所纳入的论文根据以下 3 个标准进行分类。

1)主题相关性:所选文献必须具体针对与 IIoT 相关的软件缺陷数据中的预测技术和方法,确保研究与 SDP 中的实际应用紧密相关。

2)研究质量:论文必须来源于具有良好声誉的学术期刊和会议,这些期刊和会议实施严格的同行评审流程,反映出科学界在 IIoT 安全性和效率方面的最新见解。

3)时效性:研究涵盖了 2018 年至 2025 年间发表的文献,捕捉 SDP 领域的前沿发展,确保所引用的信息既具有前瞻性又富有创新性。

2.1.2 排除标准

排除的论文主要分为以下 3 类。

1)发布类型:未发表的手稿和非同行评审材料,包括博客文章,被排除在外,以确保所有引用的作品都是经过学术审查的正式出版物。

2)语言标准:排除未用英语或中文撰写的论文,以维护研究的国际标准和学术诚信。

3)研究选择性:当多个来自同一研究团队的论文针对同一主题时,仅纳入最具代表性或最初发表的作品,以防止冗余和过度引用。

2.2 数据特征提取

本节旨在通过严格的质量控制审查,深入探讨与工业物联网软件相关的文献,以提升综述的准确性。对筛选文献进行了多维度的解读与分析,有效组织工业物联网软件缺陷预测领域的知识。本文将数据特征归纳为以下几个方面:作者、年份、训练方法、实验结果、数据集、应用场景以及局限性。

综上所述,表 2 列举了最终分析中包含的 61 篇文章,其中 2018—2020 年间的文章占比不到 25%,而约 75%的文章覆盖了 2021—2025 年的时间段,从而确保了综述的时效性。

表 2 所选文献年份概览

Table 2 Summary of articles divided by years

年份	文章数	比例/%
2018	2	3.28
2019	10	16.39
2020	3	4.92
2021	10	16.39
2022	10	16.39
2023	16	26.23
2024	9	14.75
2025	1	1.64

2.3 缺陷预测评价指标

软件在工业物联网开发中需具备稳定性和可靠性。主要评估指标包括准确率、召回率、F1 分数、AUC 分数、平均绝对误差和均方误差等,从而优化预测模型,降低软件故障率,确保系统的高效稳定运行。

2.3.1 准确率

在分类问题中,准确率(Precision)是评价一个分类器性能的重要指标,其计算式如下:

$$Precision=\frac{TP}{TP+FP} \tag{1}$$

其中，TP 表示真实正例(True Positive)，指分类器将正类样本正确地预测为正类的数量，TN 表示真实负例，指分类器将负类样本正确地预测为负类的数量，FP 表示假正例，指分类器将负类样本错误地预测为正类的数量，FN 表示假负例，指分类器将正类样本错误地预测为负类的数量。

准确率的值在 0 和 1 之间，越接近 1 的值代表分类器预测的准确度越高。

2.3.2　召回率

在面向工业物联网软件的缺陷预测领域，召回率(Recall)也常被称作真正例率(True Positive Rate，TPR)或敏感性(Sensitivity)。召回率能够衡量模型正确检测到异常数据的能力，即在所有异常数据中，被模型成功识别出的比例。其计算式如下：

$$Recall=\frac{TP}{TP+FN} \tag{2}$$

其中，TP(True Positive)表示模型检测到的真实异常数，FN(False Negative)表示模型未检测到的真实异常数。较高的召回率意味着模型能够准确地捕捉到更多的异常样本，从而降低漏报(即未检测到异常)的风险。

2.3.3　F1 分数

为了追求高准确率，模型可能将绝大多数数据点分类为正常，这会导致许多真实的异常点被遗漏，因此召回率较低。相反，若模型为了提高召回率而偏向于将数据点判断为异常，虽然能捕获更多真正的异常，但同时增加了假阳性的数量，即将正常点误判为异常，从而降低了准确率。F1 分数是基于准确率(Precision)和召回率(Recall)两个指标进行综合考虑的，用于评估分类模型在正负样本分类情况下的综合性能。其计算式如下：

$$F1\text{-}Score=\frac{2\times(Precision\times Recall)}{Precision+Recall} \tag{3}$$

F1 分数的取值范围在 0 到 1 之间，其中 1 表示完美的分类器，0 表示最差的分类器。F1 分数在准确率和召回率中找到了一个平衡点，使得分类器在精确率和召回率之间取得良好的性能。

2.3.4　AUC 分数

AUC 分数(Area Under the Curve)作为基于 ROC 曲线(Receiver Operating Characteristic Curve)的性能评估指标，AUC 分数能够综合考虑模型在不同阈值设置下的真正例率与假正例率，从而有效评价模型在极端类别不平衡条件下的总体性能。ROC 曲线是通过将分类器的不同阈值下的假阳性率(FPR)和真阳性率(TPR)绘制在二维平面上得到的曲线。其中 FPR 的计算式如下：

$$FPR=\frac{FP}{FP+TN} \tag{4}$$

其中，FP 是假阳性的数量，TN 是真阴性的数量，TP 是真阳性的数量，FN 是假阴性的数量。AUC 分数不仅考虑了分类器的精确率，也考虑了召回率，可以从全局角度反映出分类器的性能。

2.3.5　平均绝对误差

平均绝对误差(Mean Absolute Error，MAE)通过计算模型预测出的缺陷数量与实际发生的缺陷数量之间偏差的平均值，来评估模型预测的准确性。这一指标直观反映了预测结果与真实情况的平均接近程度，是评价模型预测性能优劣的有效手段。平均绝对误差指模型预测值$\hat{y}_i$与样本真实值y_i之间距离的平均值。其计算式如下：

$$MAE=\frac{1}{m}\sum_{i=1}^{m}|y_i-\hat{y}_i| \tag{5}$$

其中，$y_i=\{y_1,y_2,y_3,\cdots\}$为样本真实值，$\hat{y}_i=\{\hat{y}_1,\hat{y}_2,\hat{y}_3,\cdots\}$为样本预测值，$m$ 为样本的个数。MAE 提供了预测误差的平均绝对量度，由于是绝对值，避免了正负误差相抵消的情况，MAE 的值越小，表示预测模型的准确度越高，预测结果与实际数据的偏差越小，但 MAE 对所有类型的误差都给予相同的权重，即它不区分大误差和小误差。

2.3.6　均方误差

均方误差(Mean Squared Error，MSE)能够体现预测缺陷数与实际缺陷数之间的差异程度，是一种刻画模型预测值$\hat{y}_i$与观测数据真实值y_i之间偏差平方的平均值的统计度量。其计算式如下：

$$MSE=\frac{1}{m}\sum_{i=1}^{m}(y_i-\hat{y}_i)^2 \tag{6}$$

其中，$y_i=\{y_1,y_2,y_3,\cdots\}$为样本真实值，$\hat{y}_i=\{\hat{y}_1,\hat{y}_2,\hat{y}_3,\cdots\}$为样本预测值，$m$ 为样本的个数，MSE 的值越接近于 0 表示模型预测越准确。然而，MSE 容易受到异常值的影响，因为较大误差会被平方放大。这意味着 MSE 对模型的异常预测非常敏感，即模型对单个或少数数据点的大错误会显著增加 MSE 的值。

3　文献综述

3.1　文章分类

图 1 给出了本文中 SDP 研究的分类法，分为 3 个主要领域：统计方法、机器学习和模型导向的方法。

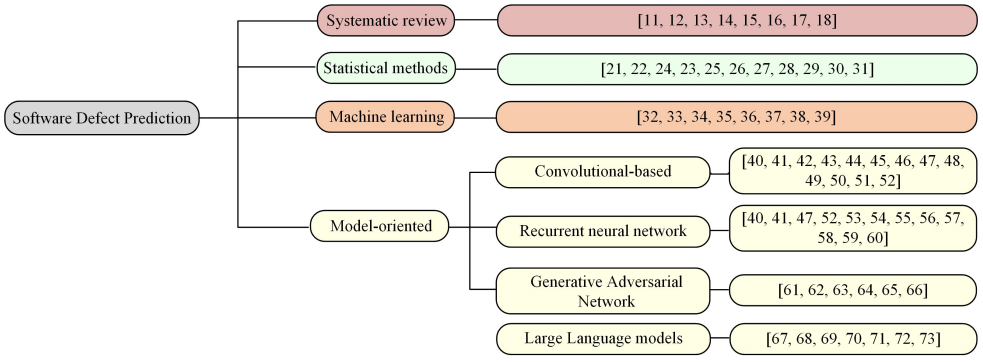

图 1　软件缺陷预测的分类

Fig. 1　Taxonomy of software defect prediction

1)统计方法主要通过历史数据分析来进行趋势预测；2)在机器学习领域，评估了多种基本算法，但未包括深度学习技术；3)在深度学习范畴内，针对模型导向的方法进行了评估，这些方法包括卷积神经网络(Convolutional Neural Network，CNN)、循环神经网络(Recurrent Neural Network，RNN)、生成对抗网络(Generative Adversarial Network，GAN)以及大型语言模型(Large Language Models，LLMs)。

表3列出了时间数据分析中3种重要方法的权衡。

表3 分类方法的优点和局限性

Table 3 Advantages and limitations of classification methods

方法	类型	优点	缺点
传统方法	Average	简单易用	忽略代码特征
	Regression	可解释性强	仅适用线性关系
机器学习	—	自动特征提取	特征工程依赖
深度学习	CNN	捕捉局部代码模式	需大量训练样本
	RNN	时序处理优	梯度消失问题
	GAN	生成缺陷样本增强数据	训练不稳定
	LLMs	理解代码语义	需微调且计算成本高

统计方法在处理静态和线性模式方面表现出优越的性能，但在应对非线性动态时却面临挑战。机器学习方法虽然擅长自动特征提取，但通常需要大量的特征工程。相比之下，深度学习方法(如CNN，RNN，GAN和LLMs)则促进了自动特征学习。尽管它们在捕获局部和全局上下文方面的效率有所不同，但通常伴随着较高的计算成本。

3.2 基于统计学的方法

表4综合概述了相关的研究成果，通过对现有软件操作数据的深入分析，统计学方法能够协助预见软件系统潜在的缺陷发展趋势，从而提升工业物联网软件系统的可靠性和稳定性。基于统计学的方法在工业物联网软件缺陷预测领域的应用主要涵盖参数估计和假设检验两个步骤。在参数估计阶段，通常采用最大似然估计[19]和贝叶斯估计[20]等方法对数据进行模型构建并得到其参数分布。在假设检验阶段，通过构造合理的假设，比较样本观测值与模型预测值之间的差异来评估软件缺陷的概率，以实现对软件缺陷的及时准确预测。

多种统计学方法显著提升了预测的准确性与效率。Al-omari等[21]引入的学习排名方法，旨在通过优先级排序帮助质量保证团队识别高风险模块。实验结果表明，该方法在多个数据集上超越传统分类方法，能够有效提高资源的分配效率。然而，其局限性在于不同数据集之间的精度差异和数据噪声的影响，可能导致一致性结果的获取较为困难。相较之下，Sivavelu等[22]开发的分段一致性回归索引极限学习分类器在工业物联网场景中展示了突出的故障预测能力，尤其在精度、召回率和时间复杂度方面表现优异。尽管如此，该方法也可能受到输入数据质量的影响，从而影响其广泛适用性。

表4 统计学方法文献概览表

Table 4 Overview table of statistical methods literature

作者	方法	应用场景	数据集	应用场景
Al-omari等[21]	监督	LTR方法在预测易发生缺陷的模块方面优于传统分类方法，提供更好的排名	PROMISE	软件模块优先级排序
Sureka等[22]	监督	PRILEC在精度、召回率和时间复杂度方面显著提高	PROMISE	工业物联网故障预测
Li等[25]	监督	比较不同模型在预测滑坡易感性方面的表现，强调识别高风险区域的重要性	滑坡数据集	灾害管理与规划
Khan等[23]	监督	提出的迁移学习方法在软件故障预测方面有显著改进	3个相关项目	跨项目故障预测
Wang等[24]	监督	基于包装的子集评估器性能最佳，逻辑回归模型实现最高性能	4个现实世界项目	新版本故障预测
Singh等[27]	监督	机器学习模型的表现优于传统统计方法，PNN性能超过BPN	NASA	实时系统故障监控
Cui等[28]	监督	KSSDP模型在8个数据集中表现优异，准确率达96.7%	NASA	开源软件缺陷管理
Suryawanshi等[29]	监督	结合逻辑回归与梯度下降的模型在缺陷预测中实现良好准确性	CM1缺陷数据集	缺陷分类研究
Pritchard等[30]	监督	三阶段模型表现出更好的预测能力，优于传统方法	未指定特定数据集	软件测试优化
Rathore等[26]	监督	核心回归方法表现最佳，强调数据质量的重要性	PROMISE	历史数据故障预测
Yu等[31]	监督	回归算法整体实现较低的AAE和较高的Pred值，但在缺陷模块中表现较差	PROMISE	模块测试资源分配

在跨项目故障预测方面，Khan等[23]提出的迁移学习方法通过整合多个项目的数据显著提升了预测的准确性。Wang等[24]则通过特征选择技术，发现基于包装的子集评估器在选择最佳预测变量方面表现最佳，同时逻辑回归模型在故障预测中实现了最高的性能。这些方法有效提高了新版本故障预测的准确性和可扩展性，但仍需注意数据特征的选择与模型的适应性。

在灾害管理与规划领域，Li等[25]强调了识别滑坡易发区域的重要性，推动了决策支持的进步。Rathore等[26]研究了多种回归方法，指出核心回归方法在预测软件系统故障方面表现最佳，特别强调了数据质量对模型性能的影响。Singh等[27]展示的机器学习模型在实时系统故障监控中表现出色，尤其是概率神经网络(Probabilistic Neural Network，PNN)较传统统计方法具有更高的准确性。Cui等[28]开发的KSSDP模型在8个数据集上达到96.7%的准确率。相对而言，Suryawanshi等[29]结合逻辑回归与梯度下降的方法在缺陷分类研究中取得了良好准确性，而Pritchard等[30]的三阶段调整回归模型在软件测试优化中表现更优。最后，Yu等[31]建议在模块测试资源分配中合理使用回归算法进行模块排名，尽管在某些模块中可能表现不佳。

3.3 基于机器学习的方法

本节探讨采用机器学习方法进行IIoT回归测试优化、软件缺陷识别、跨版本缺陷预测以及关键系统故障预测的相关技术与应用。表5综合概述了相关的研究成果，需要强调的是，本章不涉及基于深度学习技术的分类，因为这些分类在机器学习中代表了一种独特的方法。相关的分类将会在后续章节中进行全面讨论。

表 5　基于机器学习方法的文献概览表

Table 5　Literature overview table based on machine learning

参考文献	训练方式	实验结果	数据集	应用场景
Habtemariam 等[32]	无监督	在平均故障检测百分比方面优于随机技术,结果不依赖于世代数	3 个基准程序	回归测试优化
Sreedevi 等[33]	监督	MMDBM 算法在性能和准确率上优于现有方法	NASA,MDP	软件缺陷识别
Xu 等[34]	半监督	HALKP 框架实现的 F 测量、g 均值和均衡值分别为 0.480,0.592 和 0.580,优于基准方法	MORPH 数据集	跨版本缺陷预测
Shen 等[35]	监督	贝叶斯优化随机森林模型在缺陷预测方面表现更好,召回率除外	NASA,MDP	软件缺陷预测
Wei 等[36]	监督	LTSA-SVM 模型提高了 1～4%的预测精度和 F 测量	未指定数据集	IoT 系统缺陷预测
Balaram 等[37]	监督	集成随机森林显示出更高的灵敏度和特异性,优于决策树和 KNN	Camel,Ant,Jedit,log4j	关键系统故障预测
Goyal[38]	监督	FILTER 技术显著提高了 SVM 模型的性能,准确度提高 16.73%	PROMISE	软件开发质量提升
Zhang 等[39]	监督	提出的模型在 NASA 和 PROMISE 数据集中实现了高达 0.923 的 AUC 值,显示特征融合方法的有效性	NASA PROMISE	银行与医疗软件缺陷管理

在回归测试优化和软件缺陷识别的领域,采用机器学习方法可以显著提升测试的效率与准确性。Habtemariam 等[32]提出的基于遗传算法的测试用例优先级排序方法,在平均故障检测百分比方面优于随机技术,特别适合于回归测试的优化,且其结果不受世代数的影响,但需要关注参数设置的敏感性。相较之下,Sreedevi 等[33]的 MMDBM 算法通过结合数值和分类属性来增强缺陷预测的准确性。这些方法在软件缺陷识别和回归测试中的应用显示了其高效性和可扩展性。

在跨版本缺陷预测和关键系统故障预测的应用场景中,Xu 等[34]提出的 HALKP 框架结合混合主动学习和内核主成分分析,在 F-measure、均值和均衡值上的表现优于基准方法,适用于识别新版本中的潜在缺陷。Shen 等[35]利用贝叶斯优化随机森林模型在缺陷预测上表现出色,尽管在召回率上有所不足。与此同时,Wei 等[36]的 LTSA-SVM 模型在物联网系统的缺陷预测中提高了 1%～4%的预测精度,而 Balaram 等[37]通过集成随机森林与自适应合成采样技术,成功提升了灵敏度和特异性。Goyal[38]的 FILTER 技术显著提高了 SVM 模型的性能,准确度提升达 16.73%。最后,Zhang 等[39]结合混合采样和多目标优化,为 IIoT 的软件缺陷管理提供了有效解决方案,在 NASA 和 PROMISE 数据集上实现了高达 0.923 的 AUC 值。

3.4　基于模型导向的方法

为有效地对模型导向的方法进行分类,首先需要识别出不同的架构范式。各项研究已提出基于基础机制的分类,包括基于 CNN,RNN,GAN 和 LLMs。

3.4.1　基于卷积神经网络的方法

CNN 由多个不同的层组成,包括用于特征提取的卷积层、用于减少空间维度和降低过拟合的池化层、用于整合特征以进行决策的全连接层。表 6 综合概述了相关的研究成果,在软件缺陷预测中,CNN 能够有效挖掘代码特征和历史数据中的潜在模式,通过卷积层提取出关键特征,并利用池化层缩减过多的信息,从而提高缺陷预测的准确性。

表 6　基于卷积神经网络方法的文献概览表

Table 6　Literature overview table based on convolutional neural network

参考文献	训练方式	实验结果	数据集	应用场景
Zhang[40]	监督	深度学习模型在预测精度和召回率上显著优于传统方法	未具体说明	软件缺陷预测与质量保证
Lin 等[44]	监督	LEDNet 在分类正常和有缺陷芯片方面实现 5.04%的低不准确率	LEDC 缺陷生成数据库	LED 芯片质量检测
Zhang 等[41]	监督	利用基于注意机制的 CNN 和 BiLSTM 从中提取丰富而独特的特征集	未具体说明	软件缺陷检测与管理
Liu 等[43]	无监督	基于 CNN 的缺陷预测方法在 5 个软件项目中实现 0.707 1 的平均 F1 分数,集中测试资源于缺陷率最高模块	PROMISE	早期缺陷发现与修复
Huang 等[42]	监督	基于重子节点的 AST 方法在 F1 值和 AUC 值上平均提升约 3%和 4%	Apache	软件缺陷预测精度提升
Khleel 等[47]	监督	CNN 和 GRU 结合的模型在平衡数据集上的 AUC 提高了 19%,平均精度达到 90%	PROMISE	项目内缺陷预测
Balasubramaniam 等[51]	监督	CNN SALO 模型在精度和准确性上优于 SVM 和 BiLSTM 等现有方案	NASA 数据集	测试资源分配
Qiu 等[45]	监督	基于 CNN 的缺陷预测技术在预测精度上明显优于传统方法	未具体说明	软件维护与演进
Zeng 等[46]	监督	Gcn2Defect 模型的 F1 得分为 0.792,优于多个基准模型	PROMISE	依赖网络预测软件缺陷
Qiu 等[48]	监督	CNN-THFL 框架在 72 项跨项目缺陷预测任务中表现优于其他方法	PROMISE	跨项目缺陷预测
Sekaran 等[49]	监督	ECNN 模型在精度和 F 测量方面优于其他模型	CDA	项目内软件缺陷预测
Li 等[52]	监督	在预测软件缺陷方面表现有效,使用多种性能指标进行评估	SARD	别名和漏洞检测
Nevendra 等[50]	监督	在 14 个数据集中优于决策树回归和随机森林回归等算法	PROMISE	版本内和跨版本缺陷预测

在软件缺陷预测与质量保证领域，基于深度学习的方法展现出显著的优势。Zhang[40]和 Zhang 等[41]提出的层次特征集成深度学习方法，结合了抽象语法树（Abstract Syntax Tree，AST）、类依赖网络（Class Dependency Network，CDN）及传统功能，利用注意力机制的 CNN 和双向长短期记忆网络（Bidirectional Long Short-Term Memory Network，BILSTM）提取独特特征。尽管此方法理论基础扎实，但在捕捉深层语义和语法信息方面仍存在不足，可能影响特征整合的全面性。相比之下，Huang 等[42]提出的基于重子节点的 AST 方法，通过保留节点类型和语义信息，提升了预测精度，尽管实施时间成本较高，但在 F1 值和 AUC 值上显著优于传统方法。Liu 等[43]利用 CNN 实现了 0.707 1 的平均 F1 得分，有效聚焦于高缺陷率模块，加速了问题解决。

在特定应用场景中，Lin 等[44]将 CNN 与类激活映射（Class Activation Mapping，CAM）结合，成功降低了 LED 芯片质量检测中的缺陷检测不准确率，确保了产品质量。同时，Qiu 等[45]在软件维护与演进中显著提升了潜在缺陷的识别能力，展示了方法的实际应用效果。Zeng 等[46]提出的 Gcn2Defect 模型通过依赖网络分析获得了 0.792 的 F1 得分，凸显了其在多项目中的适用性和准确性。此外，Khleel 等[47]结合 CNN 与门控循环单元（Gated Recurrent Unit，GRU），实现了 92%的平均精度，成功解决了类别不平衡问题，进一步提升了模型的预测能力。

针对跨项目缺陷预测，Qiu 等[48]开发的 CNN-THFL 框架在 72 项任务中的表现突出，显示了其处理项目间数据分布的适应性。Sekaran 等[49]提出的增强卷积神经网络（ECNN）展现出更高的精度和 F 测量性能，而 Nevendra 等[50]基于指标的卷积神经网络（MB-CNN）在 14 个数据集中优于其他基准，显示出其在版本内和跨版本缺陷预测中的有效性。Balasubramaniam 等[51]通过优化 CNN 模型，结合增强的主成分分析技术提升了缺陷检测的准确性，Li 等[52]则通过特征融合与别名分析展示了新方法用于检测程序漏洞的过程。

3.4.2 基于循环神经网络的方法

在软件缺陷预测中，RNN 能够有效地分析代码变更历史、开发过程中的时间序列数据以及其他相关的上下文信息，从而提高缺陷预测的准确性和可靠性。

表 7 列出了 RNN 相关的研究成果，Zhang[40]通过结合 CNN 和 BiLSTM，提出的深度学习模型能够有效处理大规模和高维数据，显著提升了预测的精度和召回率，成为软件缺陷预测与质量保证的重要工具。Khleel 等[53]则采用了 BiLSTM 网络与过采样技术，有效应对数据不平衡问题，研究结果表明，使用 SMOTE 等技术能够显著提升模型在准确性和召回率。

表 7 基于循环神经网络方法的文献概览表

Table 7 Literature overview table based on recurrent neural network

参考文献	训练方式	实验结果	数据集	应用场景
Zhang[40]	监督	在预测精度和召回率上显著优于传统方法	未具体说明	缺陷预测与质量保证
Khleel 等[53]	监督	使用平衡数据集提升了预测结果，比较了多项性能指标	PROMISE	时间依赖捕获
Zhang 等[41]	监督	方法旨在提升预测模型性能，具体实验结果未详细列出	未具体说明	缺陷检测与管理
Pandey 等[54]	监督	JITCP-Predictor 在 MCC 和 F-Measure 上明显优于 6 种基线方法	crossdata	软件变更故障检测
Li 等[52]	监督	框架在缺陷预测方面有效，使用多种性能指标进行评估	SARD	不可推断别名和漏洞检测
Fan 等[55]	监督	DP-ARNN 在 F1-measure 上平均提高了 14%，在 AUC 上提高了 7%	Apache	缺陷识别
Dam 等[56]	无监督	模型在随机森林的 AUC 为 0.98，逻辑回归的召回率为 0.86	PROMISE	潜在源代码缺陷识别
Borandag[57]	监督	RNN 算法在准确性和其他性能指标上优于传统机器学习算法	SFPXP-TDD	早期的故障预测
Khleel 等[47]	监督	CNN 和 GRU 结合的模型在平衡数据集上的 AUC 分别提高了 19%和 24%	PROMISE	项目内缺陷预测
Wang 等[58]	监督	GH-LSTM 在 F-measure 和 IFA 等指标方面的表现优于现有方法	PROMISE	测试资源分配与软件质量提升
Munir 等[59]	监督	DP-AGL 模型在召回率、精度等指标上的表现优于传统方法	Code4Bench	软件测试与代码审查
Farid 等[60]	监督	CBIL 模型的 F-measure 为 0.852，高于其他测试模型	PROMISE	软件缺陷自动识别

在特征组合与模型融合方面，Zhang 等[41]通过引入基于注意机制的 CNN 和 BiLSTM，将 AST 与 CDN 相结合，有效提升了特征的丰富性和独特性，使得模型在处理复杂数据时能够更好地捕捉关键信息。Pandey 等[54]开发的 JITCP-Predictor 模型在跨项目数据合成方面表现优越，明显优于多种基线方法，显示出跨项目学习的潜力。Li 等[52]通过将程序解析为 AST 和程序依赖关系图，结合 LSTM 和图卷积网络，展现了该方法在不可推断别名和漏洞检测中的有效性。

在模型性能的提升方面，Fan 等[55]提出的 DP-ARNN 模型通过注意力机制，平均提升了 F1-measure 和 AUC，成功识别了软件维护过程中的潜在缺陷。Dam 等[56]提出的深层树模型结合 LSTM 结构与 AST 匹配，在源代码潜在缺陷识别中表现突出。Borandag[57]结合 RNN 与集成学习，强调了深度学习方法在软件生命周期早期故障预测中的应用价值。此外，Khleel 等[47]进一步结合 CNN 和 GRU，通过 SMOTE Tomek 方法有效处理类别不平衡问题，显著提高了模型在平衡数据集上的表现。Wang 等[58]提出的 GH-LSTM 模型在 F1-measure 和 IFA 等指标上的表现优于现有方法，为测试资源分配与软件质量提升提供了支持。Munir 等[59]开发的 DP-AGL 框架利用基于注意力的 GRU-LSTM 架构，在软件测试与代码审查场景中展示了提升的性能。最后，Farid 等[60]提出的 CBIL 模型结合 CNN 和 BiLSTM，实现了软件缺陷的自动识别。

3.4.3 基于对抗神经网络的方法

表 8 综合概述了 GAN 相关的研究成果，GAN 由两个神经网络组成，分别是生成器和判别器。生成器从随机噪声中生成假数据，而判别器的任务是区分真实数据和生成的数据。这两个网络通过对抗训练来实现相互竞争：生成器不断改进以生成更真实的数据，而判别器则不断提高识别能力。最终，训练的目标是使生成器生成的数据足够逼真，以至于判别器

无法准确区分真假数据。

基于 GAN 的方法逐渐成为解决数据不平衡问题和提高预测准确性的有效工具。Luo 等[61]提出的 SPC-GAN 模型通过矢量编码和生成语义等效的代码段,在实时软件缺陷检测中显著提高了 F1 分数和检测性能,特别是在开发人员提交代码更改时表现突出。然而,研究指出语义保留的变化对非生成模型产生显著影响,表明在处理复杂语义变更时,该模型仍需进一步优化。Chouhan 等[62]则采用 GAN 进行老化相关错误预测,实验结果表明,其在处理老化相关软件缺陷时,显著提升了模型性能,验证了 GAN 在特定任务中的有效性。

表 8　基于对抗神经网络方法的文献概览表

Table 8　Literature overview table based on generative adversarial network

参考文献	训练方式	实验结果	数据集	应用场景
Luo 等[61]	无监督	SPC-GAN 模型在 F1 分数上显著改善,与传统模型相比提高了检测性能	Apache	实时检测代码更改中的缺陷
Chouhan 等[62]	无监督	GAN 在老化相关错误预测模型中的性能提升显著	ARB	预测软件中的老化相关错误
Kumar 等[63]	无监督	使用 NASA,MDP 数据集时,GAN 的准确度提高了 0.8%、精度提高了 2.2%、召回率提高了 2.3%、F1 分数提高了 3.2%	NASA,MDP	数据不平衡项目的缺陷预测
Pal[64]	无监督	基于 GAN 的方法在跨项目缺陷预测中表现良好,但因类别不平衡影响了模型性能	JDT	跨项目缺陷预测
Xu 等[65]	无监督	与传统方法相比,SDP 聚合模型的 F1 平均值提升显著	PROMISE	类不平衡情况
Zhang 等[66]	无监督	在 26 个不平衡数据集上的精度、召回率、F-measure 和 G-mean 均优于传统过采样方法	NASA,AEEEM	类不平衡情况

此外,Kumar 等[63]利用 NASA,MDP 数据集证明,通过生成合成数据来改善实例不平衡问题,分类指标如准确度、精度和 F1 分数均有所提升,尽管这种改进幅度不大,但仍展示了 GAN 在应对数据不平衡问题时的潜力。Pal[64]的研究则聚焦于跨项目缺陷预测,尽管基于 GAN 的方法能够提高预测准确性,但类别不平衡问题仍会影响模型的整体性能,凸显了跨项目学习中的数据质量挑战。Xu 等[65]和 Zhang 等[66]均提出了基于 GAN 的聚合和过采样方法,在不平衡数据集上进行实验,证明了这些方法在精度、召回率及 F1 分数等多个指标上明显优于传统方法,为软件缺陷预测提供了更为可靠的解决方案。

3.4.4　基于大语言模型的方法

LLMs 可以分为仅编码模型和仅解码模型。仅编码模型以 BERT 架构为代表,被广泛应用于缺陷分析和特征提取。相对而言,仅解码模型以 GPT 架构为例,在自回归序列生成方面表现出色。虽然仅编码模型在捕捉序列中的上下文信息方面表现优异,但仅解码模型则专门为生成连贯的序列而设计。

如表 9 所列,Song 等[67]提出的 DP-GANPT 模型采用半监督学习,结合自然语言和编程语言的双模态输入,显著提高了项目内缺陷预测(Within-Project Defect Prediction,WPDP)和跨项目缺陷预测(Cross-Project Defect Prediction,CPDP)的 F1 分数,分别达到 62.3 和 54.6,超越了多种最先进的模型。然而,该模型依赖于有限的标记样本和特定的 PROMISE 数据集,可能会限制其在其他编程语言和项目中的推广能力。与之相似,Mao 等[68]开发的 SDP-SLAC 方法的 F-measure 在多个开源 Java 项目中比传统方法提高了 27.9%,但主要集中于 Java。

表 9　基于大语言模型的文献概览表

Table 9　Literature overview table based on large language models

参考文献	训练方式	实验结果	数据集	应用场景
Song 等[67]	半监督	DP-GANPT 在 WPDP 中的平均 F1 分数为 62.3,在 CPDP 中最高为 54.6,优于多种 SOTA 模型	PROMISE	缺陷模块识别
Mao 等[68]	监督	SDP-SLAC 在多个 Java 项目上的 F-measure 比 PROMISE-DP 提升了 27.9%	PROMISE	Java 项目
Qu 等[69]	监督	在 8 种缺陷类型上实现 80% 以上的准确率,部分类型在 60%～80%之间	SARD	代码缺陷检测的效率
Mashhadi 等[70]	监督	CodeBERT 在多项指标上相比传统模型提高了 29 到 140	综合数据集	自动预测源代码错误
Sultan 等[71]	监督	未提供具体实验结果,指出计算复杂性和编码风格的挑战	综合数据集	关注空指针取消引用
Nakhla 等[72]	无监督	LLM4FL 的 Top-1 准确性在 Defects4J 基准测试中提升了 19.27%	Defects4J	复杂系统的故障定位
Hossain 等[73]	监督	Toggle 在 CodexGlue 和 Defects4J 上实现优越性能,超越了其他方法的多项指标	CodexGlue,Defects4J	自动错误本地化与修复

此外,Qu 等[69]结合图结构与深度神经网络,成功实现了 80%以上的缺陷检测准确率,但也面临函数与变量名称变化对精度的影响。Mashhadi 等[70]应用大型语言模型 CodeBERT,展现出在错误严重度预测方面的显著优势,改进幅度从 29 到 140 不等,显示了其在源代码分析中的潜力。Sultan 等[71]关注空指针取消引用的检测,虽然未提供具体实验结果,但指出了处理大型代码库时计算复杂性和编码风格适应性等问题。Nakhla 等[72]提出的 LLM4FL 方法在 Defects4J 基准测试中提升了 19.27%的 Top-1 准确性,适用于复杂系统的故障定位。最后,Hossain等[73]的 Toggle 框架通过分离错误本地化与修复流程,展现出在多项指标上的优越性能,显著提高了软件工程中的自动错误识别与修复效率。

4　数据集

在 IIoT 中选择与研究主题密切相关的典型数据集,对验证模型效能和确保研究成果的可信度至关重要。表 10 系统性地概述了公开数据集的关键信息,包括数据集的名称、来源地址、样本总量、特征维度以及异常样本比例,以此为研究者提供一个清晰、方便的参考框架。

表 10 公开数据集概览表

Table 10 Overview table of public dataset

数据集	来源地址	程序数	涉及语言
PROMISE	http://promise.site.uottawa.ca/SERepository/datasets-page.html	20	Java,C,C++
Apache Ant	https://cwiki.apache.org/confluence	15	Java
SIR	https://sir.csc.ncsu.edu/portal/index.php	83	Java,PHP,C#,C,C++
SARD	https://samate.nist.gov/SARD/	1	C,C++,Java
KAMEI	https://research.cs.queensu.ca/~kamei/jittse/jit.zip	6	C,C++,Java
SATE IV	https://www.nist.gov/itl/ssd/software-quality-group/static-analysis-tool-exposition-sate-iv	1	C,C++
NASA	https://github.com/klainfo/NASADefectDataset	13	Java,PHP,C#,C,C++

具体而言,PROMISE 数据集[74]由 He 等收集,涵盖多个软件工程项目的数据,包括代码质量、缺陷报告与修复历史,为软件缺陷预测提供了重要的实证基础。在嵌入式软件和 IIoT 系统中,其分析结果对于预测软件缺陷、优化系统性能及提升设备间互操作性至关重要。相比之下,Apache Ant 数据集是通过 Apache 基金会多个项目的实际运行与测试而获取的,包含用户行为日志、系统性能指标及代码变更历史,提供了多维度数据,有助于嵌入式软件系统的缺陷预测。ARB 数据集[75]汇集了 Linux Kernel 与 MySQL 这两个大型开源软件项目的相关数据,该数据集特别针对业务关键系统和安全关键系统中软件的老化相关缺陷,通过应用软件复杂性度量进行预测分析。同时,SIR 数据集提供了多种编程语言(如 Java,PHP 等)的软件工件,涵盖从源代码到测试用例的广泛信息,为分析软件质量提供了基础数据,但其有效性需结合具体应用进行评估。

此外,SARD 数据集作为软件漏洞测试的权威资源,由美国国家标准与技术研究院(NIST)维护,包含多种已知漏洞类型,极大支持了嵌入式软件的安全性分析和缺陷预测。KAMEI 数据集[76]提供多种缺陷度量元,能够量化软件缺陷概率,为机器学习技术的应用提供支持。NASA 数据集[77]包含多个用于软件指标分析的子数据集,详细记录静态代码指标及缺陷数据,为缺陷预测研究提供了丰富的信息。而 MNIST[78],CIFAR-10[79],ImageNet[80]和 SATE IV[81]等数据集在视觉识别和静态分析工具性能评估方面具有独特价值,为嵌入式系统的缺陷预测提供了重要的实验依据。

5 未来发展趋势

5.1 复杂异构环境下的缺陷模式动态变化

IIoT 的高度异构性体现在设备层(如传感器、PLC、边缘计算节点)和通信层(如 MQTT、OPC UA 等多协议共存),这种异构架构导致缺陷模式呈现出动态演化的特征,其变化规律受硬件配置、网络状态和实时性需求等多维因素的共同影响。传统基于静态假设的分析方法(如线性回归、卷积神经网络)难以有效建模这种动态特性,而现有的序列建模方法(如 RNN 及其变体)在长周期依赖关系建模方面仍然存在显著局限性。融合图神经网络(Graph Neural Network,GNN)与时空注意力机制的动态图建模方法(如 Graph Transformer)[82]能够有效表征 IIoT 系统中的设备拓扑关系和数据流的时空特征[83],通过联合分析设备状态矩阵和通信日志张量,可以显著提升对分布式系统缺陷(包括数据不同步、边缘节点失效等典型故障)的预测准确率。

5.2 数据稀缺与标注成本高昂

缺陷样本的极端稀疏性与标注过程对领域专家(如设备运维工程师)的高度依赖,使得传统的数据增强方法(如生成对抗网络)所生成的合成数据往往缺乏物理可实现性,难以满足工业场景的真实性要求。采用领域自适应的大语言模型[84-85](如 LLaMA-3 的工业微调版本),通过精心设计的提示工程,可以准确模拟 PLC 梯形图中的典型逻辑缺陷(如死锁条件、竞态条件等)。进一步结合数字孪生技术构建的虚拟仿真环境能够生成具有物理真实性的带标签故障日志,这种基于物理模型的增强方法为解决小样本学习问题提供了新的技术路径。

5.3 实时性与资源受限的冲突

边缘设备(如基于 ARM 架构的现场网关)所面临的有限计算资源与缺陷预测任务对实时性(毫秒级响应)的严格要求之间存在显著矛盾。当前主流的大规模语言模型(如 BERT)和深度神经网络(如 ResNet)因其高计算复杂度而难以直接在资源受限的边缘设备上部署。为了解决这一问题,可以构建轻量化的大模型与神经符号系统的协同架构。一方面,通过采用混合专家系统(MoE)[86-87],或利用知识蒸馏技术(如 TinyLlama)实现模型的压缩与优化[88]。另一方面,建立神经符号联合推理框架,其中大语言模型负责处理非结构化日志数据,而符号推理系统[89]负责验证可解释的缺陷逻辑规则(例如工业通信协议冲突检测)。这种混合架构在确保模型性能的同时,显著降低了对计算资源的需求,实现了资源利用与预测精度之间的有效平衡。

5.4 模型可解释性与用户认知理解

虽然深度学习具有卓越的预测性能,但高维非线性黑盒模型由于其内部机制对终端用户而言缺乏透明度和可解释性,这直接影响了基于模型输出的决策可信度。将大语言模型(如 GPT-4、BERT)的强大理解能力与可解释性工具(如 LIME[90],SHAP[91])相结合,不仅实现了高精度缺陷检测,还能明确识别影响结果的关键因素。通过增强对模型决策原理的理解,这种方法有望推动发展更可靠、以用户为中心的工业物联网缺陷预测系统。具体而言,可采用注意力机制可视化技术来解析模型关注的关键代码片段,并结合因果推理方法区分真实缺陷与系统正常波动。

结束语 本文对 2018—2025 年在 IIoT 领域发表的关于 SDP 的研究文章进行了系统分析,重点关注模型、算法、数据集、应用场景和性能评估等各类主题,总结了不同方法论所面临的挑战,特别强调了以模型为导向的主要方法,包括 CNN,RNN,GAN 以及 LLMs。未来的研究应优先关注复杂异构环境下缺陷模式的动态变化,解决数据稀缺和标注成本高昂的问题,同时平衡实时性与资源限制的矛盾。此外,还应增强模型的可解释性和用户的认知理解,以提升系统的可理解性和操作的鲁棒性。

参 考 文 献

[1] STRANDBERG P E,AFZAL W,SUNDMARK D. Software test results exploration and visualization with continuous integration and nightly testing[J]. International Journal on Software Tools for Technology Transfer,2022,24(2):261-285.

[2] AL QASEM O,AKOUR M,ALENEZIM. The influence of deep learning algorithms factors in software fault prediction[J]. IEEE Access,2020,8:63945-63960.

[3] BHANDARI K,KUMAR K,SANGALA L. Data quality issues in software fault prediction:a systematic literature review[J]. Artificial Intelligence Review,2023,56(8):7839-7908.

[4] PANDEY S,KUMAR K. Software Fault Prediction for Imbalanced Data: A Survey on Recent Developments[J]. Procedia Computer Science,2023,218:1815-1824.

[5] SRIVASTAVA V,CHAUHAN R K,LOHIA P. Highly efficient cesium-based halide perovskite solar cell using SCAPS-1D software:theoretical study[J]. Journal of Optics,2023,52(3):1218-1225.

[6] DEVALE P,JADHAV R B,BIDWE R V,et al. Machine Learning and Just-in-Time Strategies for Effective Bug Tracking in Software Development[J]. International Journal of Intelligent Systems and Applications in Engineering,2024,12(6s):749-758.

[7] ZHENG W,SHEN T,CHEN X,et al. Interpretability application of the Just-in-Time software defect prediction model[J]. Journal of Systems and Software,2022,188:111245.

[8] GOYAL S. Static code metrics-based deep learning architecture for software fault prediction[J]. Soft Computing,2022,26(24):13765-13797.

[9] SATAPATHY S C,JENA A K,SINGH J,et al. A novel approach of software fault prediction using deep learning technique [J]. Automated Software Engineering:A Deep Learning-Based Approach,2020:73-91.

[10] KHAN B,IQBAL D,BADSHAH S. Cross-Project Software Fault Prediction Using Data Leveraging Technique to Improve Software Quality[C]// Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering. 2020:434-438.

[11] WU F J. Research progress on static software defect prediction [J]. Journal of Computer Science and Exploration,2019,13(10):1621-1637.

[12] LI Y,LIU Z D,ZHANG H J. A survey on cross-project software defect prediction methods[J]. Computer Technology and Development,2020,30(3):98-103,121.

[13] LIU X T,GUO Z Q,LIU S R,et al. Comparative experiments among software defect prediction models:Issues,progress and challenges[J]. Journal of Software,2023,34(2):582-624.

[14] CAI L,FAN Y R,YAN M,et al Research progress on just-in-time software defect prediction[J]. Journal of Software,2019,30(5):1288-1307.

[15] BATOOL I,KHAN T A. Software fault prediction using deep learning techniques[J]. Software Quality Journal,2023,31:1-40.

[16] ZAIN Z M,SAKRI S,ISMAILN H A. Application of Deep Learning in Software Defect Prediction:Systematic Literature Review and Meta-analysis[J]. Information and Software Technology,2023,158:107175.

[17] PANDEY S K,MISHRA R B,TRIPATHI A K. Machine learning based methods for software fault prediction:A survey[J]. Expert Systems with Applications,2021,172:114595.

[18] RATHORE S S,KUMAR S. A study on software fault prediction techniques[J]. Artificial Intelligence Review,2019,51:255-327.

[19] LIU Y,LIU B. A modified uncertain maximum likelihood estimation with applications in uncertain statistics[J]. Communications in Statistics-Theory and Methods,2024,53(18):6649-6670.

[20] SAINSBURY-DALE M,ZAMMIT-MANGION A,RICHARDS J,et al. Neural Bayes estimators for irregular spatial data using graph neural networks[J]. Journal of Computational and Graphical Statistics,2025,34:1-16.

[21] AL-OMARI S,ELSHEIKH Y,AZZEH M. A New Learning to Rank Approach for Software Defect Prediction[J]. International Journal of Advanced Computer Science and Applications,2022,13(8).

[22] SIVAVELU S,PALANISAMY V. Piecewise Congruence Regressed Indexive Extreme Learning Classifier for Software Fault Prediction[J]. IEEE Access,12.

[23] KHAN B,IQBAL D,BADSHAH S. Cross-Project Software Fault Prediction Using Data Leveraging Technique to Improve Software Quality[C]// Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering. 2020:434-438.

[24] WANG H,KHOSHGOFTAAR T M. A study on software metric selection for software fault prediction[C]// 18th IEEE International Conference On Machine Learning And Applications(ICMLA 2019). IEEE,2019:1045-1050.

[25] LI L,NAHAYO L,HABIYAREMYE G,et al. Applicability and performance of statistical index,certain factor and frequency ratio models in mapping landslides susceptibility in Rwanda[J]. Geocarto International,2022,37(2):638-656.

[26] RATHORE S S. An exploratory analysis of regression methods for predicting faults in software systems[J]. Soft Computing,2021,25:14841-14872.

[27] SINGH R,RATHORE S S. Linear and non-linear bayesian regression methods for software fault prediction[J]. International Journal of System Assurance Engineering and Management,2022,13(4):1864-1884.

[28] HUANG X W,FAN G S,YU H Q,et al. Software defect prediction based on heavy sub－node abstract syntax trees[J]. Journal of Computer Engineering,2025,42(1):25-2,48.

[29] SURYAWANSHI R,KADAM A. Software Defect Prediction by Logistic Regression with Gradient Descent Cost Computation [C]// International Conference on Emerging Smart Computing and Informatics(ESCI)2024. IEEE,2024:1-5.

[30] PRITCHARD S,MITRA B,NAGARAJU V. Three-Stage Adjusted Regression Forecasting for Software Defect Prediction [C]// Annual Reliability and Maintainability Symposium (RAMS 2024). IEEE,2024:1-6.

[31] YU X,KEUNG J,XIAO Y,et al. Predicting the precise number of software defects:Are we there yet? [J]. Information and Software Technology,2022,146:106847.

[32] HABTEMARIAM G M,MOHAPATRA S K. A genetic algo-

rithm-based approach for test case prioritization[C]// Information and Communication Technology for Development for Africa: Second International Conference(ICT4DA 2019). Bahir Dar, Ethiopia, Revised Selected Papers 2. Springer International Publishing, 2019: 24-37.

[33] SREEDEVI E, PREMALATHA V, SIVAKUMAR S, et al. A comparative study on new classification algorithm using NASA MDP datasets for software defect detection[C]// International Conference on Intelligent Sustainable Systems (ICISS 2019). IEEE, 2019: 312-317.

[34] XU Z, LIU J, LUO X, et al. Cross-version defect prediction via hybrid active learning with kernel principal component analysis [C]// IEEE 25th international conference on software analysis, evolution and reengineering (SANER 2018). IEEE, 2018: 209-220.

[35] SHEN Y, HU S, CAI S, et al. Software Defect Prediction based on Bayesian Optimization Random Forest[C]// 9th International Conference on Dependable Systems and Their Applications(DSA 2022). IEEE, 2022: 1012-1013.

[36] WEI H, HU C, CHEN S, et al. Establishing a software defect prediction model via effective dimension reduction[J]. Information Sciences, 2019, 477: 399-409.

[37] BALARAM A, VASUNDRA S. Prediction of software fault-prone classes using ensemble random forest with adaptive synthetic sampling algorithm[J]. Automated Software Engineering, 2022, 29(1): 6.

[38] GOYAL S. Effective software defect prediction using support vector machines (SVMs) [J]. International Journal of System Assurance Engineering and Management, 2022, 13(2): 681-696.

[39] ZHANG J, LI D, WONG W E, et al. A Hybrid Sampling and Multi-Objective Optimization Approach for Enhanced Software Defect Prediction[J]. arXiv: 2410. 10046, 2024.

[40] ZHANG Y. Software defect prediction model based on deep learning[C]// International Conference on Power, Electrical Engineering, Electronics and Control(PEEEC 2024). IEEE, 2024: 954-959.

[41] ZHANG S, JIANG S, YAN Y. A Hierarchical Feature Ensemble Deep Learning Approach for Software Defect Prediction[J]. International Journal of Software Engineering and Knowledge Engineering, 2023, 33(4): 543-573.

[42] CUI M T, WU K Q, MARIANI M S. Software defect prediction based on feature extraction and Stacking ensemble learning[J]. Computer Applications and Software, 2021, 47(12): 230-235, 248.

[43] LIU C, SANOBER S, ZAMANI A S, et al. Defect Prediction Technology in Software Engineering Based on Convolutional Neural Network[J]. Security and Communication Networks, 2022, 2022(1): 5058461.

[44] LIN H, LI B, WANG X, et al. Automated defect inspection of LED chip using deep convolutional neural network[J]. Journal of Intelligent Manufacturing, 2019, 30: 2525-2534.

[45] QIU X, FAN P, REN J. Convolutional Neural Network-Based Research on Software Engineering Defect Prediction[C]// Proceedings of the 6th International Conference on Information Technologies and Electrical Engineering. 2023: 305-308.

[46] ZENG C, ZHOU C Y, LV S K, et al. Gcn2defect: Graph convolutional networks for smotetomek-based software defect prediction[C]// IEEE 32nd International Symposium on Software Reliability Engineering(ISSRE 2021). IEEE, 2021: 69-79.

[47] KHLEEL N A A, NEHÉZ K. A novel approach for software defect prediction using CNN and GRU based on SMOTE Tomek method[J]. Journal of Intelligent Information Systems, 2023, 60(3): 673-707.

[48] QIU S, LU L, CAI Z, et al. Cross-Project Defect Prediction via Transferable Deep Learning-Generated and Handcrafted Features[C]// SEKE. 2019: 431-552.

[49] SEKARAN K, ANNABEL L S P. A Deep Learning Based Model for Defect Prediction in Intra-Project Software[C]// 7th International Conference on Trends in Electronics and Informatics (ICOEI 2023). IEEE, 2023: 1148-1155.

[50] NEVENDRA M, SINGH P. Defect count prediction via metric-based convolutional neural network[J]. Neural Computing and Applications, 2021, 33(22): 15319-15344.

[51] BALASUBRAMANIAM S, GOLLAGI S G. Software defect prediction via optimal trained convolutional neural network[J]. Advances in Engineering Software, 2022, 169: 103138.

[52] LI X, ZHU Z. Software Defect Detection Based on Feature Fusion and Alias Analysis[C]// IEEE International Test Conference in Asia(ITC-Asia 2023). IEEE, 2023: 1-6.

[53] KHLEEL N A A, NEHÉZ K. Software defect prediction using a bidirectional LSTM network combined with oversampling techniques[J]. Cluster Computing, 2024, 27(3): 3615-3638.

[54] PANDEY S K, TRIPATHIA K. Cross-Project setting using Deep learning Architectures in Just-In-Time Software Fault Prediction: An Investigation[C]// IEEE/ACM International Conference on Automation of Software Test (AST 2023). IEEE, 2023: 24-34.

[55] FAN G, DIAO X, YU H, et al. Software defect prediction via attention-based recurrent neural network[J]. Scientific Programming, 2019, 2019(1): 6230953.

[56] DAM H K, PHAM T, NG S W, et al. A deep tree-based model for software defect prediction[J]. arXiv: 1802. 00921, 2018.

[57] BORANDAG E. Software fault prediction using an RNN-based deep learning approach and ensemble machine learning techniques[J]. Applied Sciences, 2023, 13(3): 1639.

[58] WANG H, ZHUANG W, ZHANG X. Software defect prediction based on gated hierarchical LSTMs[J]. IEEE Transactions on Reliability, 2021, 70(2): 711-727.

[59] MUNIR H S, REN S, MUSTAFA M, et al. Attention based GRU-LSTM for software defect prediction[J]. Plos One, 2021, 16(3): e0247444.

[60] FARID A B, FATHY E M, ELDINA S, et al. Software defect prediction using hybrid model (CBIL) of convolutional neural network(CNN) and bidirectional long short-term memory(Bi-LSTM)[J]. PeerJ Computer Science, 2021, 7: e739.

[61] LUO N, MA Y, LI J, et al. A Just-in-time Software Defect Detection Method Using Generative Adversarial Networks[C]// 4th International Conference on Electronic Communication and Artificial Intelligence(ICECAI 2023). IEEE, 2023: 37-45.

[62] CHOUHAN S S, RATHORE S S. Generative adversarial networks-based imbalance learning in software aging-related bug prediction[J]. IEEE Transactions on Reliability, 2021, 70(2): 626-642.

[63] KUMAR P S, VENKATESAN R. IMproving software defect

prediction using generative adversarial networks[J]. Int. J. Sci. Eng. Appl，2020，9：117-120.

[64] PAL S. Generative adversarial network-based cross-project fault prediction[J]. arXiv：2105. 07207，2021.

[65] XU J P，GUO X F，WANG R B，et al. Software defect prediction aggregation model based on GAN data augmentation[J]. Computer Science，2023，50(12)：24-31.

[66] ZHANG H W，JIA X Y. Oversampling method for class-imbalanced software defect prediction based on generative adversarial networks[J]. Journal of Nanjing University of Science and Technology，，2023，47(2)：174-182.

[67] SONG W，GAN L，BAO T. Software Defect Prediction via Generative Adversarial Networks and Pre-Trained Model[J]. International Journal of Advanced Computer Science & Applications，2024，15(3).

[68] MAO J E，ZHOU S J，ZHANG S Q，et al. Software defect prediction method based on attention and cost sensitivity[J]. Computer Measurement & Control，2024，32(9)：94-100.

[69] QU T，LIU W，ZHENG W，et al. Software Defect Detection Method Based on Graph Structure and Deep Neural Network [C]// International Conference on Asian Language Processing (IALP 2022). IEEE，2022：395-400.

[70] MASHHADI E，AHMADVAND H，HEMMATI H. Method-level bug severity prediction using source code metrics and LLMs[C]// IEEE 34th International Symposium on Software Reliability Engineering(ISSRE 2023). IEEE，2023：635-646.

[71] SULTAN M F，KARIM T，SHAON M S H，et al. Enhanced LLM-Based Framework for Predicting Null Pointer Dereference in Source Code[J]. arXiv：2412. 00216，2024.

[72] NAKHLA RAFI M，KIM D J，CHENT H，et al. Enhancing Fault Localization Through Ordered Code Analysis with LLM Agents and Self-Reflection[J]. arXiv：2409. 13642，2024.

[73] HOSSAIN S B，JIANG N，ZHOU Q，et al. A deep dive into large language models for automated bug localization and repair [J]. Proceedings of the ACM on Software Engineering，2024，1(FSE)：1471-1493.

[74] HE Z，PETERS F，MENZIES T，et al. Learning from open-source projects：An empirical study on defect prediction[C]// 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. IEEE，2013：45-54.

[75] COTRONEO D，NATELLA R，PIETRANTUONO R. Predicting aging-related bugs using software complexity metrics[J]. Performance Evaluation，2013，70(3)：163-178.

[76] MALHOTRA R，SHUKLA S，SAWHNEY G. Assessment of defect prediction models using machine learning techniques for object-oriented systems[C]// 5th International Conference on Reliability，Infocom Technologies and Optimization(Trends and Future Directions)(ICRITO 2016). IEEE，2016：577-583.

[77] NASA Defect Dataset[OL]. https：//github. com/klainfo/NASADefectDataset.

[78] LECUN Y. The MNIST database of handwritten digits[J/OL]. http：//yann. lecun. com/exdb/mnist/，1998.

[79] NAIRKrizhevsky，HINTON Vinod，CHRISTOPHER Geoffrey，Mike Papadakis，and Anthony Ventresque. The cifar-10 dataset [EB/OL]. http：//www. cs. toronto. edu/kriz/ cifar. html.

[80] RUSSAKOVSKY O，DENG J，SU H，et al. Imagenet large scale visual recognition challenge[J]. International Journal of Computer Vision，2015，115：211-252.

[81] OKUN V，DELAITRE A，BLACK P E. Report on the static analysis tool exposition(sate) iv[J]. NIST Special Publication，2013，500：297.

[82] ZHANG C，XIANG J，HAO R，et al. SGT：Aging-related bug prediction via semantic feature learning based on graph-transformer[J]. Journal of Systems and Software，2024，217：112156.

[83] LU G，LING F，LI J，et al. Graph Attention-Based Dual Enhancement for Multiview Clustering[J]. IEEE Transactions on Computational Social Systems，2025：1-10.

[84] HE T，YANG M，HU W，et al. Analysis of the Effectiveness of Large Language Model Feature in Source Code Defect Detection [C]// 3rd International Conference on Artificial Intelligence and Computer Information Technology(AICIT 2024). IEEE，2024：1-4.

[85] WEN X C，WANG X，CHEN Y，et al. Vuleval：Towards repository-level evaluation of software vulnerability detection[J]. arXiv：2404. 15596，2024.

[86] SHANKAR MISHRA A，SINGH RATHORE S. Implicit and explicit mixture of experts models for software defect prediction [J]. Software Quality Journal，2023，31(4)：1331-1368.

[87] JU E，LEE J，RYU D. DefectGRANDE：Hybrid Approach for Class Imbalance in Software Defect Prediction[C]// IEEE International Conference on Big Data and Smart Computing(BigComp 2025). IEEE，2025：90-91.

[88] LIU W，YUE Y，CHEN X，et al. SeDPGK：Semi-supervised software defect prediction with graph representation learning and knowledge distillation[J]. Information and Software Technology，2024，174：107510.

[89] BHUSHAN M，DUARTE J Á G，NEGI A，et al. An ontological knowledge-based method for handling feature model defects due to dead feature[J]. Engineering Applications of Artificial Intelligence，2024，136：109000.

[90] ASAL B，DEMIR M Ö. Enhancing Software Defect Prediction through Explainable AI：Integrating SHAP and LIME in a Voting Classifier Framework[C]// 8th International Artificial Intelligence and Data Processing Symposium(IDAP 2024). IEEE，2024：1-7.

[91] YAN Z，ZHANG L. Interpretable Wind Power Prediction：A Machine Learning Perspective Using Lightgbm and SHAP [C]// 2nd International Conference on Artificial Intelligence and Automation Control(AIAC 2024). IEEE，2024：225-229.

DENG Tao，born in 2000，postgraduate，is a member of CCF(No. Q5490G). His main research interests include software defect prediction and time series prediction.

DENG Ye，born in 2004，bachelor. Her main research interests include software engineering and smart education.