软件学报 ISSN 1000-9825, CODEN RUXUEW
2026,37(8):3144−3160 [doi: 10.13328/j.cnki.jos.007596] [CSTR: 32375.14.jos.007596]

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

面向软件测试领域知识问答的大模型评估*

陈煜磊, 聂钰格, 吴化尧

(南京大学 计算机科学与技术学院, 江苏 南京 210023)

通信作者: 吴化尧, E-mail: hywu@nju.edu.cn

摘　要: 大语言模型 (large language model, LLM) 在通用任务中已展现出卓越的性能, 但其在专业领域中的可信性、鲁棒性与可用性仍缺乏系统化评估. 以软件测试教材编写为代表性应用场景, 围绕 100 个核心测试概念与方法精心构建了 700 个测试问题, 并选取 5 个代表性 LLM, 系统评估了其在阅读理解、问答及文本生成方面的能力. 实验结果表明, LLM 在大多数问题上整体表现优良, 在答案的准确性、完整性和流畅性方面均达到较高水准; 然而, 在涉及研究现状与复杂概念时, 仍存在幻觉与推理偏差等可靠性问题. 进一步分析显示, 大模型生成的内容在知识覆盖度与教育性上较传统教材具有较为明显的优势, 能够为软件测试教材的修订与教学提供有效支持. 不仅系统揭示了 LLM 在专业领域知识处理中的具体能力边界与典型缺陷, 也为基于问答驱动的智能化评估方法在专业教育与应用中的推广提供了实证依据与方法参考.

关键词: 软件测试; 知识问答; 大语言模型; 性能评估

中图法分类号: TP311

中文引用格式: 陈煜磊, 聂钰格, 吴化尧. 面向软件测试领域知识问答的大模型评估. 软件学报, 2026, 37(8): 3144–3160. http://www.jos.org.cn/1000-9825/7596.htm

英文引用格式: Chen YL, Nie YG, Wu HY. Benchmarking Large Language Models for Software Testing Knowledge Q&A. Ruan Jian Xue Bao/Journal of Software, 2026, 37(8): 3144–3160 (in Chinese). http://www.jos.org.cn/1000-9825/7596.htm

Benchmarking Large Language Models for Software Testing Knowledge Q&A

CHEN Yu-Lei, NIE Yu-Ge, WU Hua-Yao

(School of Computer Science and Technology, Nanjing University, Nanjing 210023, China)

Abstract: Large language models (LLMs) have demonstrated remarkable performance in general tasks. However, their trustworthiness, robustness, and applicability in specialized domains remain insufficiently assessed. Using the compilation of software testing textbooks as a representative application scenario, this study constructs 700 carefully designed test questions covering 100 core testing concepts and methods and systematically assesses five representative LLMs in terms of reading comprehension, question-answering (Q&A), and text generation. The experimental results indicate that LLMs generally exhibit strong performance on most questions, achieving high levels of accuracy, completeness, and fluency. However, issues of reliability, such as hallucination and reasoning bias, persist, particularly when addressing current research trends and complex concepts. Further analysis reveals that LLM-generated content provides broader knowledge coverage and greater educational value compared with traditional textbooks, offering effective support for revising and teaching software testing materials. This study not only delineates the specific capability boundaries and typical deficiencies of LLMs in processing domain knowledge but also provides empirical evidence and methodological insights for advancing Q&A-driven intelligent evaluation in professional education and applications.

Key words: software testing; knowledge question-answering (Q&A); large language model (LLM); performance evaluation

* 基金项目: 国家自然科学基金 (62472209); 江苏省重点研发计划 (BE2023025-2)
本文由“大模型与软件工程”专题特约编辑聂长海教授、王璐副教授、王莹教授、姜艳杰副教授、张敏灵教授推荐.
收稿时间: 2025-09-08; 修改时间: 2025-10-28; 采用时间: 2025-12-08; jos 在线出版时间: 2025-12-24
CNKI 网络首发时间: 2026-02-04

1 引 言

大语言模型 (large language model, LLM) 作为自然语言处理 (natural language processing, NLP) 和人工智能 (artificial intelligence, AI) 领域的突破性技术, 近年来迅速崛起, 并在各类任务中展现出卓越的性能. 这些模型不仅在文本生成、信息综合和复杂问题解决方面表现出色, 还正在深刻改变各行各业的工作流程. 以 ChatGPT、DeepSeek 等为代表的 LLM 应用, 已成为推动通用人工智能 (artificial general intelligence, AGI) 发展的关键力量. 通过其先进的自然语言处理能力, LLM 为自动化、创新和复杂决策等任务提供了前所未有的潜力, 使其成为学术界和工业界不可或缺的工具.

随着 LLM 在科学研究、工程和医疗保健等不同专业领域的广泛应用, 其在特定领域的可信性、鲁棒性和可用性问题日益成为关注的焦点. 这些问题是 LLM 得以广泛应用和持续发展的基础. 特别是在软件测试领域, LLM 的应用潜力尤为显著. 软件测试作为软件工程的关键环节, 涵盖了从单元测试生成、测试预言生成到缺陷分析等众多任务, 每种任务都需要高度的领域知识和语境推理能力. 近年来, 已有数十种基于 LLM 的软件测试新实践涌现, 例如单元测试生成[1]、测试输入生成[2]、缺陷分析[3]以及与变异测试[4]、差分测试[5]、静态分析[6]等技术的结合. 这些实践表明, LLM 有望成为推动软件测试自动化和智能化的重要力量.

然而, 最近出现的一些可信性问题和 LLM 中表现出的问题 (如鲁棒性和幻觉) 已经引起了广泛关注, 如果不能妥善解决这些问题, LLM 的广泛应用在实践中可能会受到极大阻碍. LLM 测试与评价正在成为一个专门的新方向, 是推动 LLM 和其他人工智能模型取得成功的一个重要领域. 现有方法不足以彻底评估 LLM 的真正能力, 这为 LLM 测试与评估未来研究带来了巨大的挑战和新的机遇. 大模型评测的价值和意义主要体现在性能评估、透明性、公平性和偏见检测、安全性、应用指导、学术研究、优化算法和技术等方面, 大模型评测不仅是衡量其性能的手段, 更是推动技术进步、确保公平与安全的基础. 随着大模型在各领域广泛应用, 评测的价值和意义愈发重要, 大模型评测已经成为推动 AI 技术发展、保障应用安全性和促进公平竞争的重要工具.

目前, 正值大语言模型快速发展时期, 其核心能力就是对文本数据的阅读理解、回答问题和文本生成能力. 但是当前对 LLM 的评估主要依赖于标准基准测试, 这些基准通常侧重于评估模型在生成或推理客观事实方面的性能. 虽然这些基准对于理解模型的基本能力具有重要价值, 但它们往往无法充分反映 LLM 在现实世界、特定领域应用中的复杂性和细微差别. 为了弥补这一差距, 本研究通过分析 5 个流行的大语言模型对 100 个软件测试方法的 7 个关键问题的回应, 全面评估其在准确性、完整性、流畅性和创新性等方面的表现. 通过对软件测试专业知识掌握良好的专家对模型输出文本质量进行评估, 本研究旨在揭示 LLM 在软件测试领域中的优势与局限性, 进而阐明其在该领域的可用性.

本研究中测试方法和关键问题的选择源自 2013 年出版的软件测试领域的经典教材《软件测试的概念与方法》[7]一书. 十几年前, 撰写一本专业领域知识教材需要专业人员耗费大量时间和精力, 翻阅海量文献. 而如今, LLM 的出现是否能够为实际工作和学习赋能, 提升工作效率, 很大程度上取决于其可信性、可靠性和可用性. 因此, 我们尝试利用 LLM 赋能修订编写工作, 并在此过程中回答 5 个研究问题: 1) 大语言模型能否生成正确的、可用的概念文本? 2) 不同大语言模型在概念文本生成能力上有何区别? 3) 能否基于大语言模型构建更好的专业领域教材? 4) 能否通过问答总结出大语言模型在领域知识问答上的缺陷? 如何针对这些缺陷进行改进? 5) 大模型辅助教材编写的可行路径是什么?

本研究以软件测试这一专业领域为例, 试图回答这些问题. 通过深入探讨 LLM 在软件测试中的应用潜力, 我们希望能够为 LLM 在其他专业领域的应用提供借鉴, 推动其在更广泛场景中的可信、可靠和高效使用. 本文贡献主要包括以下 3 个方面.

1) 针对软件测试领域 100 个概念和方法, 每个概念和方法设计了 7 个问题, 利用现有的最有代表性的 5 个大模型应用进行了系统的问答实验, 产生了一系列对软件测试教学、教材编写和修订有价值的结果.

2) 利用 1) 产生的结果, 根据正确性、完整性、流畅性和创新性等 4 个度量指标, 针对 LLM 的可用性、可信性和鲁棒性等重要特性, 通过评审员进行人工评估, 发现了一系列有价值的问题.

3) 建设软件测试领域知识问答的测试基准集, 以应对进一步讨论评估和应用 LLM 的未来挑战. 开源并维护

LLM 评估的相关材料, 网址为 https://gist.nju.edu.cn/llm-evaluation/, 从而培育一个软件测试概念与方法协作社区, 以进行更好的教育和评估.

本文第 2 节介绍大语言模型的相关基础及其存在问题, 介绍软件测试与大模型之间的相互作用. 第 3 节介绍实验设计和研究问题. 第 4 节展示实验结果及结论. 第 5 节介绍本文的有效性威胁. 第 6 节是相关工作及比较. 最后是本文结论及未来工作.

2 背 景

2.1 大语言模型

LLM 是一种基于深度学习的人工智能模型, 专门用于理解和生成自然语言. 与传统自然语言处理模型相比, LLM 在参数规模 (通常达到数百亿甚至万亿级)、训练语料 (跨越多领域、覆盖大规模文本)、任务泛化能力 (支持零样本和少样本学习) 以及文本生成与知识推理能力等方面具有显著优势. 这些特征使其不仅能够完成传统模型的分类、匹配等判别任务, 还能在开放域对话、知识问答和文本创作等更复杂场景中展现出前所未有的性能. 常见的 LLM 包括 OpenAI 的 GPT 系列、Google 的 BERT 和 T5 等. 它们在处理复杂的语言任务时表现出色, 广泛应用于聊天机器人、搜索引擎、内容创作等领域[8]. 近年来, 中国自主研发的大语言模型取得了显著进展. 以深度求索和通义千问等为代表的模型展示了低成本、高性能兼顾的大模型发展路径, 引发了学界和业界的广泛关注.

LLM 在现代技术中扮演着越来越重要的角色, 主要原因包括: 在自然语言理解方面, LLM 能够理解和处理人类语言中的复杂性, 帮助计算机更好地理解用户的意图和背景. 在自动化与效率方面, LLM 可以自动执行许多语言相关的任务, 如客服对话、内容生成和翻译, 从而节省时间和人力成本. 在人机交互方面, 提升人机交互的质量, 使用户可以用更自然的方式与机器进行交流, 例如通过语音或文本进行对话. 在知识获取和处理方面, LLM 能够处理和总结大量信息, 帮助用户快速获取所需的知识和信息, 特别是在信息爆炸的时代. 在个性化服务方面, LLM 可以根据用户的输入和偏好提供个性化的内容和建议, 提高用户体验. 而且可以提供多语言支持, LLM 支持多种语言, 使得全球用户都能够使用技术产品, 而不受语言障碍的限制. 支持创意和内容生成, 在创作领域, LLM 可以帮助生成创意内容, 如文章、故事、音乐或代码, 促进创新. 赋能研究与开发, LLM 在科学研究中能提供快速的信息检索和分析, 促进各领域的发展. 总体来说, LLM 提供了强大的工具和能力, 使得人们在各种场景中能够更方便高效地与机器进行交互和获得信息[9].

LLM 在近年来迅速崛起, 彻底改变了各行各业的工作流程. 通过利用先进的自然语言处理能力, 这些模型为自动化、创新和复杂决策任务提供了前所未有的潜力. 它们生成类似人类的文本、综合信息和解决问题的能力, 使其成为学术界和工业界不可或缺的工具. 随着 LLM 在专业领域 (包括科学研究、工程和医疗保健) 的日益采用, 评估其特定领域知识和可用性至关重要[10].

自 ChatGPT 从 2022 年 10 月问世以来, 关于大模型的研究变得炙手可热起来, 模型评估已经成为大模型研究的一个重要方向, 大模型的评估方面发表的文章呈上升趋势, 越来越多的研究着眼于设计更科学、更好度量、更准确的评估方式来对大模型的能力进行更深入的了解.

2.2 利用大语言模型服务软件测试 (LLM4SoftwareTesting)

在各行业与学术界的共同关注下, LLM 作为人工智能领域的一场新革命, 已在多个领域的自然语言处理任务中展现出可与人类相媲美的能力. 特别是在软件工程领域, LLM 表现出巨大潜力, 正逐步成为推动软件生命周期各关键阶段自动化与智能化变革的关键赋能者与助推器[9]. 其中, 在软件测试这一具体方向上, “LLM4Software-Testing”已成为研究热点[11]. 作为软件工程中的关键环节, 软件测试对确保软件系统符合功能、可靠性及性能要求至关重要. 该领域涵盖多种任务和上百种理论方法, 每一项任务均需深厚的领域知识和高水平的语境推理能力, 而 LLM 正在这些方面展现出广泛的应用前景.

目前, 已经出现了数十种使用 LLM 进行软件测试的新实践, 包括单元测试生成[12–14]、测试预言生成[15]、测试输入生成、缺陷分析以及与变异测试、差分测试、静态分析等各种测试技术的合作[16,17]. Li 等人[18]使用 12 个开源软件项目, 评估了 LLM 在软件测试中的有效性, 特别是在测试用例生成、错误源跟踪和错误定位方面, 研究

结果表明, LLM 可以通过检测额外的错误来有效地补充手动测试, 尽管它们容易产生幻觉, 需要仔细的人工审查. Deljouyi 等人[19]将基于搜索的软件测试和大语言模型相结合, 以提高自动生成的测试用例的可理解性. Hossain 等人[20]首次对 LLM 生成测试预言的能力进行了系统性研究, 其结果表明借助 LLM 能够有效地识别大量独特的错误. Cheng 等人[21]提出了一种基于 LLM 的输入空间划分测试方法, 该方法使 LLM 能够理解正在测试的库 API 的代码, 并基于其理解和丰富的公共知识执行输入空间划分. Li 等人[22]给出了一个端到端的解决方案, 用于自动生成 Rust 项目的单元测试. Schäfer 等人[13]提出了 TestPilot, 用于从测试骨架模板、待测试函数的代码及其文档中自动生成 JavaScript 单元测试, TestPilot 在 61% 的情况下生成了具有非平凡断言的测试, 并实现了 62% 的语句覆盖率. Li 等人[23]提出了差分提示, 以高效 (即 78% 的时间) 生成故障诱导测试用例. 他们利用 ChatGPT 推断代码样本行为的能力来创建规范和目标错误程序的正确版本, 并使用它们来推导回归预言.

尽管目前仍存在幻觉生成等可靠性问题, 需结合人工审查进行结果验证, 但多项实证研究表明, LLM 能够有效补充传统测试手段、提升覆盖率、识别更多错误, 并增强测试用例的可理解性与多样性.

2.3 大语言模型的测评 (SoftwareTesting4LLM)

近年来, 大语言模型的评测逐渐成为人工智能研究中的重要方向. 已有大量工作表明, LLM 在通用基准中性能显著提升, 推动了多种应用和工业场景的智能化发展, 也标志着人工智能进入了新的阶段. 然而, 伴随模型能力增强, 其在实际使用中也暴露出一系列问题, 例如鲁棒性不足、幻觉现象、偏见与公平性风险等. 这些问题若不能得到有效解决, 将严重制约 LLM 在关键场景中的可靠落地.

针对上述挑战, 现有研究在评测 LLM 时主要集中于 3 个方面: (1) 通用基准测试, 如 MMLU[24]、BIG-Bench[25]、SuperGLUE[26]等, 用于衡量 LLM 在多任务、多学科知识和推理能力上的表现. 这些测试揭示了大模型在通识问答和逻辑推理中的强大能力, 但也暴露了其在复杂推理、事实一致性和跨任务泛化上的不足; (2) 专业领域评测, 研究者构建了医学 (PubMedQA[27])、法律 (SARA[28])、编程 (CodeX[29]) 等领域的基准, 以验证模型在特定专业知识和任务中的适用性. 结果显示模型具备一定可用性, 但在准确性、可靠性等方面仍存在差距; (3) 非功能性特征评估, 研究逐渐关注鲁棒性、公平性、可解释性与安全性等特性. 例如, 对抗性测试用于揭示模型在输入扰动下的稳定性[30], 偏见与公平性基准用于检测潜在的不当倾向[31], 而幻觉检测方法则用于衡量模型在知识一致性上的缺陷[32]. 这些研究强调了评估 LLM 不仅要看任务表现, 还必须全面考量其可信性与可用性.

综上所述, 现有的大模型评测研究在推动 LLM 能力认知方面取得了重要进展, 但仍存在一定不足. 一方面, 主流评测多集中于通用基准和少数垂直领域, 缺乏针对软件测试等专业学科的系统化探索; 另一方面, 现有研究通常聚焦于知识问答的正确性, 对鲁棒性、可信性与可用性等其他维度的考察仍不充分. 此外, 关于 LLM 在教育和教材编纂场景的适用性研究极为有限. 针对这些不足, 本文尝试以软件测试教材编写为任务背景, 提出一种基于教材知识问答的系统化评估方法, 以系统检验大模型在专业知识理解、知识应用与教学支持方面的表现, 从而为 LLM 在教育与工程实践中的应用提供新的视角.

3 实验设计

下面介绍本文的研究问题, 采用的大模型、数据集、度量指标和实验方法论.

3.1 研究问题

我们的研究问题是: 1) 大语言模型能否生成正确的、可用的概念文本? 2) 不同大语言模型在概念文本生成能力上有何区别? 3) 能否基于大语言模型构建更好的专业领域教材? 4) 能否通过问答总结出大语言模型在领域知识问答上的缺陷? 如何针对这些缺陷进行改进? 5) 大模型辅助教材编写的可行路径是什么?

3.2 大模型选择

本文选择当前市场上多个主流 LLM 进行实验, 包括最热门的商业模型 o3-mini、Claude 3.7 Sonnet, 开源社区中表现优秀的模型 DeepSeek R1、Qwen2.5: 72B、 Llama3.3: 70B, 以全面评估不同架构和参数下的模型性能, 具体见表 1.

为了确保实验的公平性和一致性, 所有问题均以统一的格式输入每个 LLM 中, 记录每个 LLM 对每个问题的

回答, 确保输出格式一致. 实验过程中, 对于闭源商业模型 (o3-mini、Claude 3.7 Sonnet), 我们通过调用官方 API 的方式进行评估; 而对于开源模型 (DeepSeek R1、Qwen2.5: 72B、Llama3.3: 70B), 则采用本地部署的方式, 在统一的硬件和软件环境下运行, 并通过 OpenAI 兼容的 API 格式进行调用, 以尽可能减少外部因素对实验结果的干扰.

表 1 本文选择的代表性大模型

模型名称	开发机构	开源状况	模型架构	参数量
o3-mini	OpenAI	闭源	稠密	N/A*
Claude 3.7 Sonnet	Anthropic	闭源	混合推理	N/A**
DeepSeek R1	深度求索	开源	混合专家	6 710亿
Qwen2.5: 72B	阿里巴巴	开源	稠密	720亿
Llama3.3: 70B	Meta	开源	稠密	700亿

注: *表示o3-mini是闭源商用模型, OpenAI官方未披露相关参数细节. **表示Claude 3.7 Sonnet是闭源商用模型, Anthropic官方未披露相关参数细节

此外, 为了保证结果的可复现和公平性, 我们将待测 LLM 的 temperature 调至 0. 这一操作被绝大多数 LLM 测试基准所采用, 因为此时模型总是选择概率最高的 token, 对于同样的输入每次都会得到相同的输出[33].

3.3 数据集

本文参考《软件测试的概念与方法》[7]选取 100 个概念和方法 (如表 2) 作为测试对象, 并对每个软件测试的概念和方法提出 7 个问题 (见表 3): 是什么? 为什么? 理论是什么? 方法是什么? 给出实例, 优缺点是什么? 有哪些相关研究? 共 700 个问题. 这 7 个问题相互之间互相补充, 共同覆盖了软件测试的知识领域.

表 2 软件测试的概念与方法

类型	编号	概念方法	编号	概念方法	编号	概念方法	编号	概念方法
各种专门的软件测试方法	CM_1	白盒测试	CM_{11}	黑盒测试	CM_{21}	静态测试	CM_{31}	动态测试
	CM_2	基于模型	CM_{12}	基于搜索	CM_{22}	基于故障	CM_{32}	基于性质
	CM_3	蜕变测试	CM_{13}	操作剖面	CM_{23}	统计测试	CM_{33}	组合测试
	CM_4	变异测试	CM_{14}	规格说明	CM_{24}	自适应	CM_{34}	随机测试
	CM_5	反随机测试	CM_{15}	自适应随机	CM_{25}	导向性随机	CM_{35}	反模型测试
	CM_6	模糊测试	CM_{16}	成分测试	CM_{26}	GUI测试	CM_{36}	结对测试
	CM_7	在线测试	CM_{17}	探索测试	CM_{27}	极限测试	CM_{37}	有限状态机
	CM_8	模型检查	CM_{18}	TTCN	CM_{28}	UML	CM_{38}	试玩测试
	CM_9	错误猜测	CM_{19}	Petri网	CM_{29}	布尔测试	CM_{39}	状态转换
	CM_{10}	图覆盖	CM_{20}	语法测试	CM_{30}	故障注入	CM_{40}	差分测试
针对各种属性和方面的软件测试方法	CM_{41}	性能	CM_{50}	压力	CM_{59}	负载	CM_{68}	容量
	CM_{42}	可靠性	CM_{51}	安全性	CM_{60}	恢复性	CM_{69}	国际化
	CM_{43}	兼容性	CM_{52}	安装	CM_{61}	协议测试	CM_{70}	容错性
	CM_{44}	授权测试	CM_{53}	卸载	CM_{62}	备份	CM_{71}	在线帮助
	CM_{45}	鲁棒性	CM_{54}	可用性	CM_{63}	功能性	CM_{72}	数据转换
	CM_{46}	配置测试	CM_{55}	可访问性	CM_{64}	接口测试	CM_{73}	内存泄漏
	CM_{47}	一致性	CM_{56}	稳定性	CM_{65}	老化测试	CM_{74}	穿越测试
	CM_{48}	余量测试	CM_{57}	文档测试	CM_{66}	工作流	CM_{75}	数据库
	CM_{49}	存储测试	CM_{58}	输入测试	CM_{67}	领域测试	CM_{76}	使用测试
各个开发阶段的软件测试方法	CM_{77}	单元测试	CM_{79}	冒烟测试	CM_{81}	集成测试	CM_{83}	系统测试
	CM_{78}	验收测试	CM_{80}	回归测试	CM_{82}	Alpha测试	CM_{84}	Beta测试
面向各种类型软件的软件测试方法	CM_{85}	面向对象	CM_{89}	面向方面	CM_{93}	面向服务	CM_{97}	构件软件
	CM_{86}	并行并发	CM_{90}	实时系统	CM_{94}	物联网	CM_{98}	云原生
	CM_{87}	Web应用	CM_{91}	移动APP	CM_{95}	人工智能	CM_{99}	区块链
	CM_{88}	大模型LLM	CM_{92}	元宇宙	CM_{96}	开源系统	CM_{100}	嵌入式

表 3 针对每种概念和方法提出的 7 个问题

序号	问题 (prompt)	预期回答内容
P_1	是什么? (what)	介绍方法的基本概念
P_2	为什么? (why)	方法的目的, 即它可用来测试什么?
P_3	理论是什么? (theory)	介绍方法原理, 理论基础, 为什么要用这种方法?
P_4	方法是什么? (method)	具体介绍这个方法的内容, 即这个方法是什么?
P_5	给出实例 (example)	通过例子将该方法描述得更清楚
P_6	优缺点是什么? (pros and cons)	评论方法的长处和不足
P_7	有哪些相关研究? (studies)	介绍研究现状, 为学者和研究人员提供参考

3.4 度量指标

具体的度量指标我们主要采用了表 4 中的 4 个方面: 准确性、完整性、流畅性、创新性. 度量采用人工度量方式, 通过专家阅读大模型给出的答案, 从以上 4 个方面进行相互比较和打分.

表 4 度量评价指标及其说明

评价指标名称	评价维度	问题描述
准确性 (Accuracy)	事实性问题回答能力, 领域知识	模型提供的信息是否与事实一致? 未包含错误或误导性内容?
完整性 (Completeness)	知识总结能力	模型生成的内容是否全面覆盖了问题的核心要点?
流畅性 (Fluency)	语言质量	模型的回答是否自然流畅、易于理解?
创新性 (Innovation)	知识提炼与升华	模型的回答是否提供了优于基线的内容?

3.5 实验方法

针对本文的研究问题, 我们设计实验流程如图 1 所示. 整个过程可以分为 4 步: (1) 提示词构建, (2) LLM 互动, (3) 结果提取, (4) 评估与汇总.

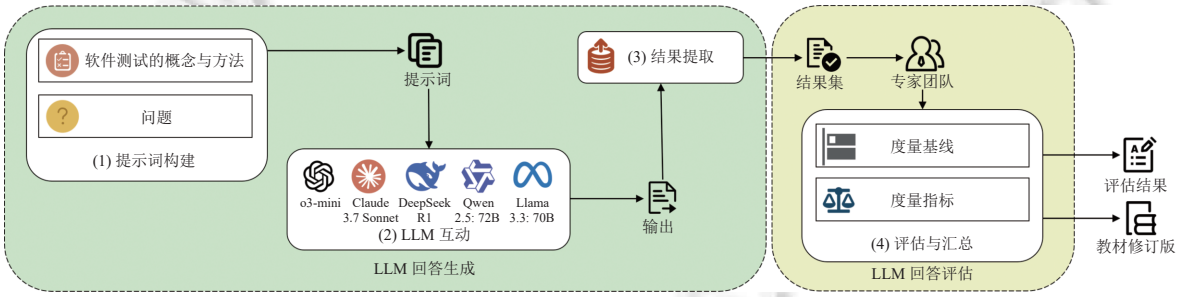

图 1 实验流程图

首先, 在提示词构建阶段, 我们从表 2 所列的 100 个软件测试概念与方法中逐一选取目标 (如“组合测试”), 并依据表 3 预设的 7 类问题 (P_1–P_7) 构建提示词, 涵盖其定义、目的、理论、方法、实例、优缺点及研究现状等方面 (如 P_1 “组合测试是什么?”、P_2 “为什么需要组合测试?”、P_3 “组合测试的理论是什么?”、P_4 “组合测试的方法是什么?”、P_5 “组合测试的实例”、P_6 “组合测试方法的优缺点是什么?”、P_7 “组合测试的相关研究如何?”).

接着, 进入 LLM 互动阶段. 将每个概念对应的 7 个问题依次输入至表 1 中的 5 个大语言模型, 由其分别生成回答. 该过程重复执行, 直至完成全部 100 个概念, 共产生 5 (模型)×7 (问题)×100 (概念)=3500 个答案.

随后是结果提取阶段. 系统收集所有模型输出, 并按问题与模型两个维度进行结构化整理, 为后续评估提供数据基础.

最后, 在评估与汇总阶段, 使用差分测试方法, 将教材内容作为基线, 对不同模型对同一问题的回答进行评分比较. 将这些模型产生的输出交给专家, 对照表 4 中提供的度量指标进行评估. 因此, 每位专家共计完成 4 (指标)×5 (模型)×7 (问题)×100 (概念)=14000 次评估. 评分过程中, 每个答案均由多位专家共同讨论并直接达成一致分数. 其中, 评分标准为若出现事实性错误、关键信息遗漏或表述不清等问题, 则依据其严重程度进行扣分; 若无上述缺陷, 则给予满分. 同时, 专家对照教材架构, 进行人工整合, 完成教材的修订工作.

针对以上实验流程, 我们可以采集由四元组{概念和方法名称 CM_i (i∈[1, 100]), 问题 P_j (j∈[1, 7]), 大模型名称 A_k (k∈[1, 5]), 度量指标 M_l (l∈[1, 4])}所描述的数据格式进行分析比较.

4 实验结果及分析

以下根据实验结果和分析, 分别回答第 3 节提出的 5 个研究问题.

1) 大语言模型能否生成正确的、可用的概念文本?

图 2 显示了 o3-mini、Claude 3.7 Sonnet、DeepSeek R1、Qwen2.5: 72B 和 Llama3.3: 70B 生成的所有回答在正确性 (M1)、完整性 (M2)、流畅度 (M3) 和创造性 (M4) 等 4 个度量上的平均分. 结果表明所有模型在各度量上的平均分均在 4.79 分以上 (满分 5 分). 其中, 表现最好的 DeepSeek R1 平均分高达 4.87 分, 即便是排名最后的 Llama3.3: 70B, 其平均分也达到了 4.80 分. 较高的整体评分表明, 模型生成的文本在绝大多数情况下是令人满意的. 各模型的评分标准差控制在 0.3–0.5 范围内, 也显示了各模型的生成质量具有良好的稳定性和可预测性.

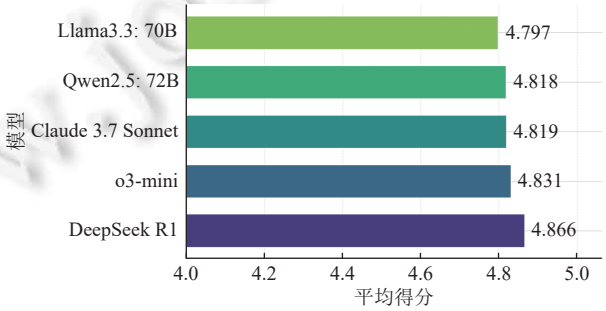

图 2 模型整体表现平均分

对多数软件测试概念, 各大语言模型都能处理得当. 尤其是对于如单元测试、集成测试、系统测试等概念, 所有模型几乎都能给出接近满分的回答. 这些高分概念通常具备明确的评估标准和预期结果, 且有成熟的实践和案例支持. 以数据转换测试为例, 它涉及明确的输入到输出转换规则, 易于形式化描述, 结果验证直观. 这些概念的特点使模型能够更准确地理解和生成相关内容, 并提供相关建议. 尽管如此, 模型在处理一些较为复杂或有歧义的概念 (如 Concolic 测试、极限测试), 以及涉及特定专业领域测试对象的概念 (如嵌入式系统) 时, 得分偏低, 说明其能力边界依然存在.

图 3 展示了 5 个模型在 7 类问题上的得分分布. 整体来看, 模型普遍能够较好地回答概念性和理论性问题, 但在涉及研究现状的问题上表现明显不足, 几乎没有能够生成高质量的回答. 为了进一步理解这种差异背后的规律, 我们对各问题类别的评分结果计算了 Spearman 相关系数, 以分析不同问题类型之间在模型表现趋势上的相似性. 该方法通过不同问题间的相关系数差异, 可辅助推断在不同问题类型背后依赖的模型的共同核心能力, 并有效判断模型能力的泛化性与特异性.

从图 4 的结果可以看出, 问题类别之间的相关性揭示了大模型在多种知识维度上的潜在能力结构: 例如, 优缺点、研究现状和理论问题三者呈现较高相关性, 说明模型在处理需要较强分析性和理论推演的问题时体现出一致的表现趋势; 定义与目标类问题相关性较强, 反映了模型在概念识别和解释性表达方面的共性能力; 而案例与方法

类问题则更紧密相关, 说明模型在推演和发散性生成上的能力具有一定一致性. 这种相关性分析不仅帮助我们理解模型在不同类型知识任务上的能力划分, 也为后续教材编写中问题设计的针对性提供了参考.

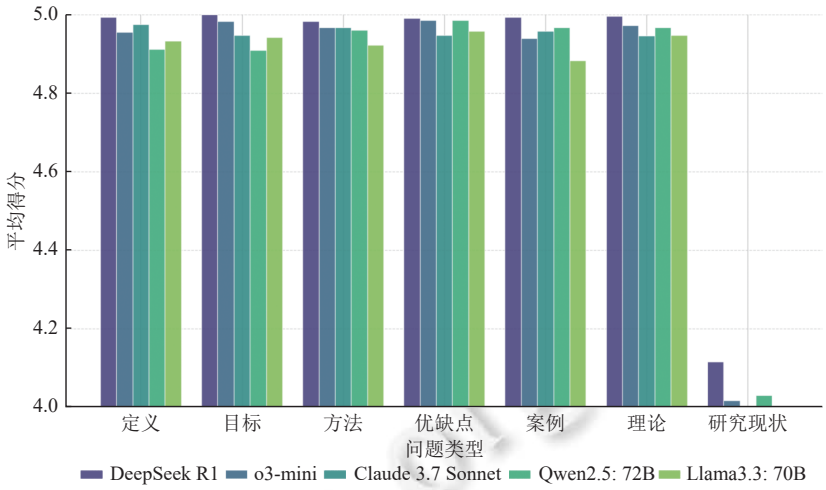

图 3 各模型在不同问题上的评分

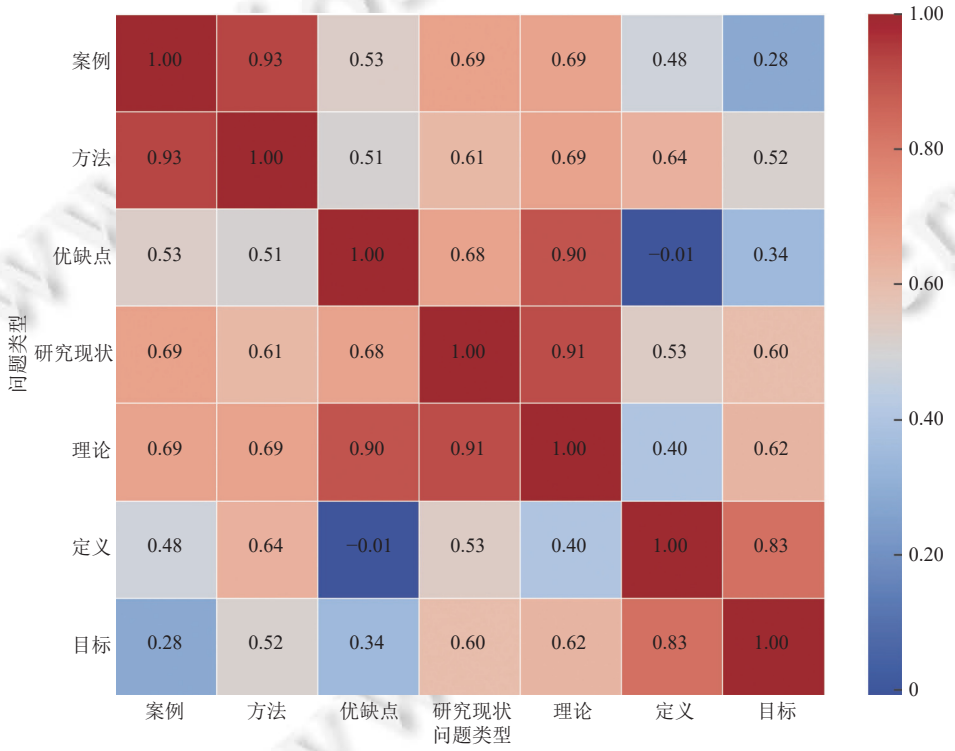

图 4 大模型在不同问题上的回答评分相关性

另一方面, 所有模型在回答研究现状类问题时均表现不佳. 这类问题往往要求模型整合最新的研究进展并给出准确的总结, 而实验数据表明模型生成的回答中幻觉比例较高, 且在引用文献或事实时存在明显的时效性不足. 这一现象揭示出两个重要问题: 一是训练数据在覆盖最新研究成果方面存在缺陷, 二是当前模型架构仍难以彻底

解决幻觉问题. 因此, 研究现状类问题成为 LLM 在教育和科研应用中最薄弱的环节, 也突出了进一步改进数据更新机制和幻觉缓解方法的重要性.

2) 不同大语言模型在概念文本生成能力上有何区别?

根据评分统计, 大语言模型普遍在概念文本生成上表现出较高的水平. 然而, 尽管所有模型都能生成基本可用的文本, 但它们在不同维度和特定任务上的表现存在一定差异. 图 5 展示了各模型在不同度量维度上的平均评分. 总的来看, 流畅性在所有模型中都获得了出色的分数, 表明近年大模型在人类偏好对齐上的巨大进步. 在准确性上, 各模型也表现出了相对较高的水平, 平均分差距微小, 这表明 LLM 作为知识库是较为可靠的. 创新性是所有模型得分最低且差异最大的维度, 体现了大语言模型在创造性思维能力上仍有一定的缺失.

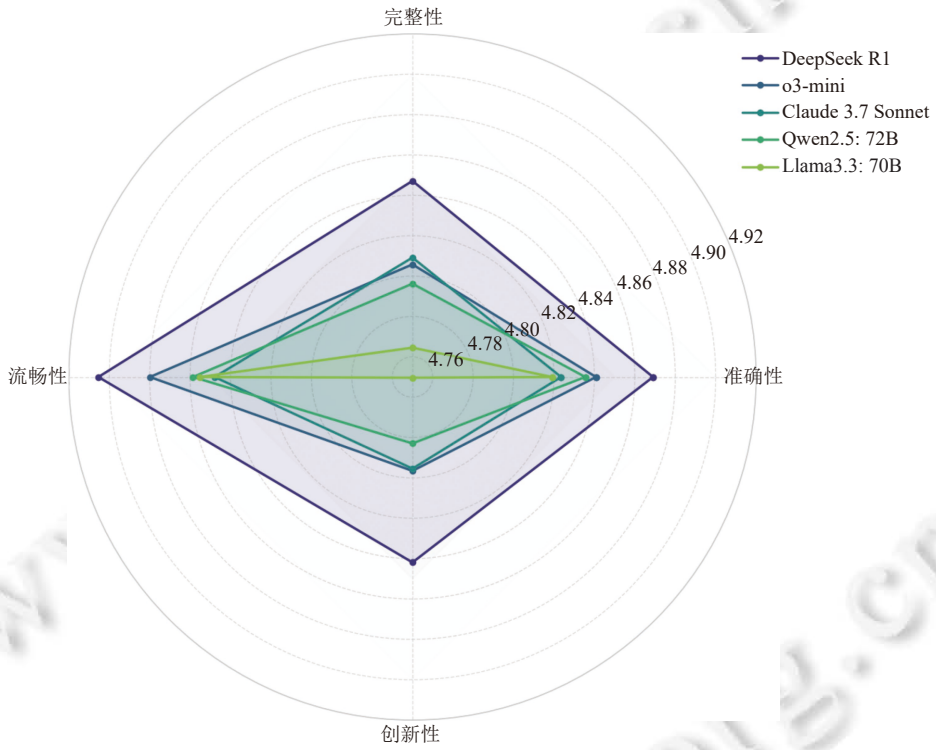

图 5 各模型在不同度量上的评分

具体来看, 模型在不同度量上的能力存在一定区分, 并呈现出一定的梯队分布. DeepSeek R1 在本次评估的所有 4 个维度中均排名第 1, 表现全面领先. o3-mini 在准确性和流畅性上表现突出, 仅次于 DeepSeek R1. 这表明其适用于对事实精度和文本可读性要求高的场景. Claude 3.7 Sonnet 的综合能力与 o3-mini 差距不显著. 其最突出的优势在于完整性, 能够提供覆盖面较广的解答. 但在流畅性方面表现较弱, 文本可能不如其他顶尖模型精炼. Qwen2.5: 72B 的综合能力与 Claude 3.7 Sonnet 相仿. 相比 Claude 3.7 Sonnet, Qwen2.5: 72B 在完整性和创新性上有一定劣势, 但在流畅性和准确性上胜出. Llama3.3: 70B 则在完整性和创新性方面表现相对其余模型更落后, 尽管它们能够生成准确的文本, 但在提供全面信息和独特视角方面能力稍弱.

对不同模型两两之间的 Wilcoxon 检验也验证了上述判断. 作为一种非参数统计检验方法, 该方法通常用于比较两个相关样本是否存在显著差异. 如图 6 所示, DeepSeek R1 相对其他模型体现了显著优势, Llama3.3: 70B 显著落后, 而其他 3 种模型基本处于同一梯队. 图 7 从不同度量进行了进一步分析. 结果显示, DeepSeek R1 各维度均具有压倒性优势, Llama3.3: 70B 在完整性和创新性上存在显著劣势, 而另外 3 个模型基本无显著差距.

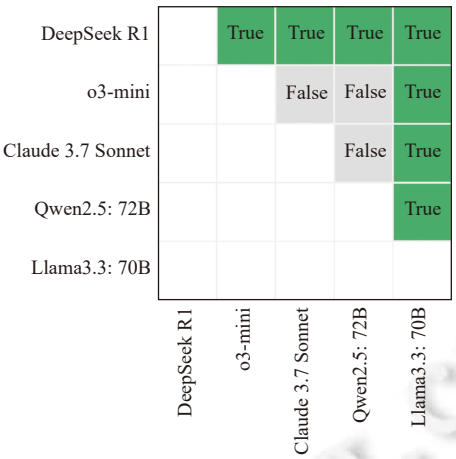

图 6　各模型整体评分的 Wilcoxon 检验结果

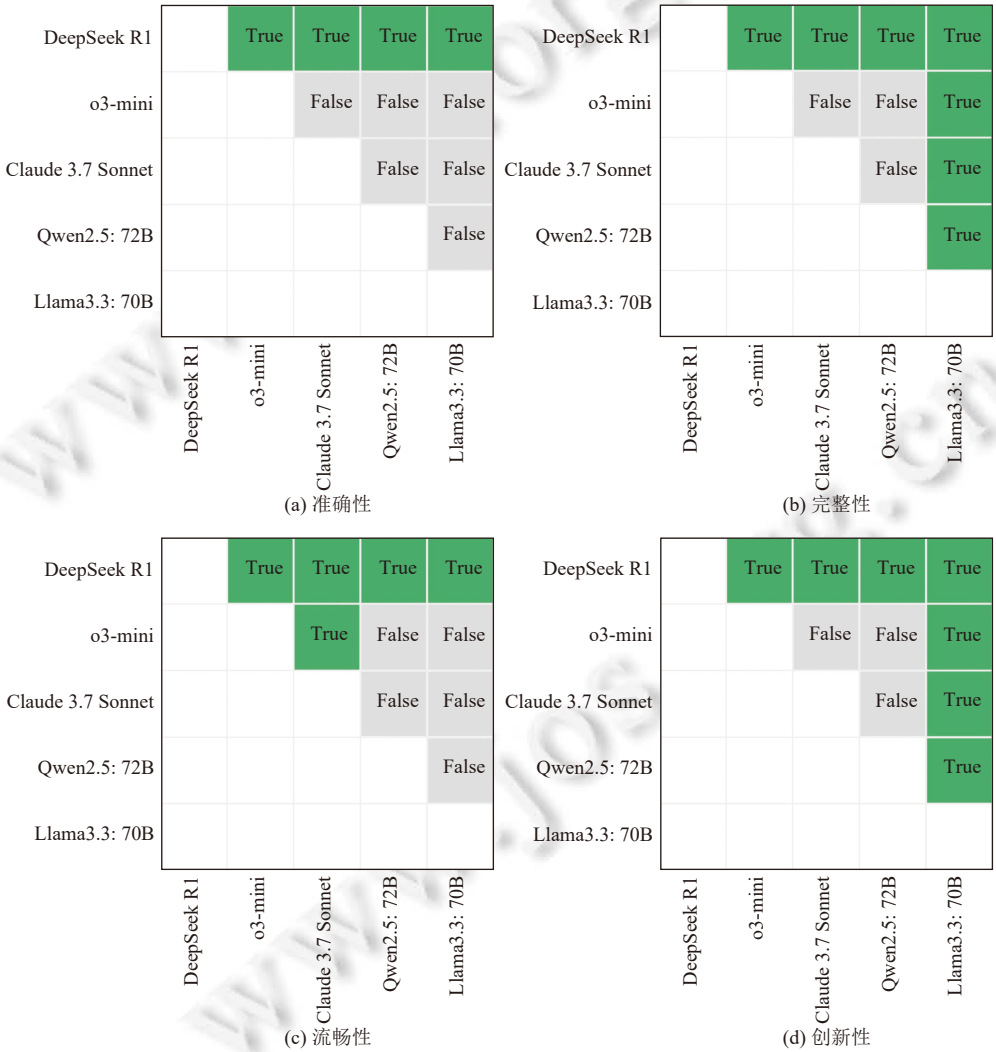

图 7　各模型在不同度量上评分的 Wilcoxon 检验结果

我们进一步对不同模型针对同一测试概念的回答情况进行分析和排名. 如图 8 所示, 在 100 个不同测试概念上, 综合能力较强的模型 DeepSeek R1 也显著体现了相对其他模型的优势, 在 72 个测试概念上取得了总分第 1. 然而, 正如研究问题 1 中所述, 所有模型平均得分普遍较高, 没有显著差距, 尤其是在其表现较为优秀的概念上. 如图 9 所示, 在 5 个模型上表现最佳的 5 个概念的平均分几乎都是满分. 但是, 不同模型在回答最差的概念上的得分显示出了一定的差距. 其中 Claude 3.7 Sonnet 在最佳前 5 和最差前 5 概念中表现出最小的性能差距, 体现其具有更高的鲁棒性, 其次是 DeepSeek R1. 我们继续提取了各模型在 10 个共同最差概念 (即在所有模型表现最差概念中重复次数排在前 10 个的概念) 上的评分. 如图 10 所示, 结果与前文的分析显示出较高一致性, Claude 3.7 Sonnet 在保持回答基本可用的情况下, 在共同最差概念上的表现最为稳定.

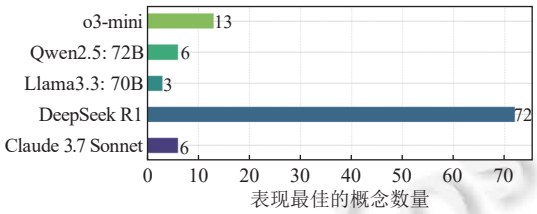

图 8 各模型在不同概念上的优势分布

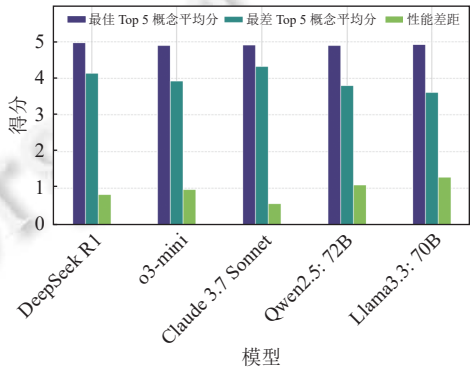

图 9 各模型在复杂和简单概念上的表现对比

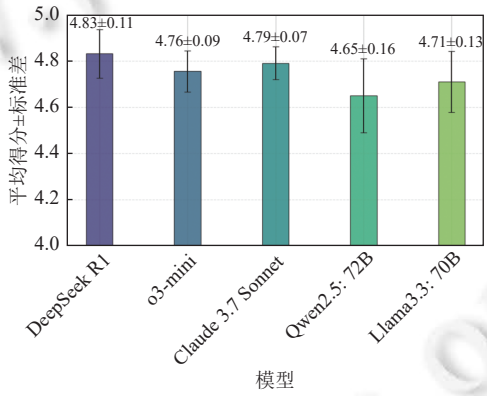

图 10 各模型在共同最差概念上的表现对比

3) 能否基于大语言模型构建更好的专业领域教材?

实验结果表明, 与传统教材相比, 5 个主流大语言模型在软件测试领域展现出显著更广的知识覆盖度. 如图 11 所示, 我们对教材中涉及的 100 个核心概念和方法 (见表 2) 进行了系统性评估, 考察其对 7 类关键问题的覆盖情况, 并与大模型生成的回答进行了对比. 结果显示, 大模型生成的内容在这些核心概念相关问题上的覆盖率比教材提升了约 270%. 现有教材在经典理论方面已具备较好的覆盖度 (大模型在该部分的覆盖率仅提升 30.7%), 而在更具时效性的内容上, 例如新兴的云系统与人工智能软件系统的测试, 大模型则表现出明显优势, 其覆盖率比教材高出 228%.

除了概念、问题覆盖率的扩展, 大模型生成内容对本书已有内容也有较大改进. 经基于大模型的评估器判断, 在教材和大模型生成中均存在的内容中, 90% 以上的大模型回答相比原教材更具优势. 可以认为大语言模型在知识广度, 内容深度, 教育性等维度上均相比原有教材体现了重大进步. 总的来说, 大模型的主要优势如下.

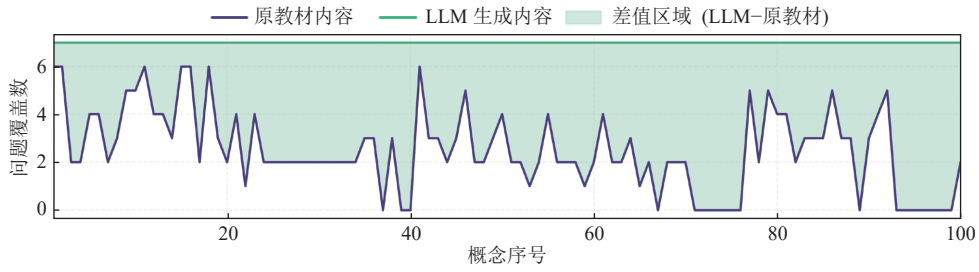

图 11 大模型生成内容与原教材内容比较

(i) 语言通俗易懂, 更具教育性. 由于教材编写者为本领域专家, 因此原教材在语言表述中往往更重专业性、严谨性, 但偏理论的表达方式往往对于初学者不友好. 而大模型则擅长使用丰富的类比与示例. 在白盒测试中, LLM 使用“给房子做体检”等类比, 将抽象的软件内部检查过程形象化, 降低了理解门槛.

(ii) 知识关联性更强, 形成系统化认知网络. 大模型在回答问题时能够动态建立跨章节、跨领域的知识联结, 显著提升了内容的综合性和迁移价值. 相较于教材相对模块化的知识呈现方式, 大模型会主动构建概念之间的逻辑桥梁. 例如在讲解“单元测试”时, 不仅解释其定义, 还会关联到“持续集成中的测试自动化”“测试覆盖率与代码质量的关系”等延伸知识点.

(iii) 更贴近现代实践, 具备实用性. 大模型回答相较于教材更注重方法在工业界当前的具体实践, 会提供具体策略、框架 (如测试包)、工具选项和场景化例子. 如在安全性测试中推荐了 Burp Suite、Nessus 等具体工具, 为读者提供了可以直接上手的实践指导.

4) 能否通过问答总结出大语言模型在领域知识问答上的缺陷? 如何针对这些缺陷进行改进?

尽管 LLM 在软件测试概念文本生成上表现出较高的质量 (平均分 4.79+), 但仍存在一些关键弱点. 首先, LLM 普遍存在幻觉问题. 在问题 P_7 “相关研究部分”中, 包括如下问题.

(i) 完全捏造. 在 LLM 知识库中没有足够信息时, 它们往往根据常见学术论文的结构, 生成看似真实的研究条目. 如在备份测试中, Qwen2.5: 72B 生成了“Evaluating Backup and Recovery Strategies in Cloud Computing”“An Empirical Study on the Effectiveness of Backup Testing Techniques”等似乎合理的条目, 但它们实际上是完全虚构的论文.

(ii) 错误关联. LLM 了解该领域的知名学者或重要文献, 但无法精确匹配时, 往往产生张冠李戴的结果. 例如 Claude 3.7 Sonnet 提到, 等价类划分是由 Glenford J. Myers 在《软件测试的艺术》中首先提出. 虽然在书和作者上没有犯错, 然而该方法却并非由 Myers 首创, 这就造成了事实性错误.

(iii) 过度泛化与模糊引用. 对于不确定来源的知识, LLM 有时会采用模糊的表述, 从而避免被证伪的风险. 如在 Beta 测试中, Qwen2.5: 72B 引用了一项来自《Journal of Product Innovation Management》的研究, 但不指出具体论文. 虽然无从指摘, 但对于寻求真实案例的用户来说, 这个回答显然是无效的.

根据上述问题总结, 幻觉问题的一个重要成因是静态训练数据中相关知识的缺失或存在偏差. 因此, 一种有效的缓解策略是使用检索增强生成技术, 从经过验证的、权威的学术数据库中检索相关文献, 辅助回答. 另一部分幻觉也暴露了模型的指令遵循不足问题. 可通过更完善的提示词设计, 明确约束模型提供真实可查证的论文信息来缓解这一问题.

另一大幻觉问题存在于复杂或歧义概念的生成中. 这暴露了 LLM 在专业概念理解上的模糊性. 如表 5 所示. 针对这些歧义概念, 也可参考幻觉问题的缓解策略, 结合更完善的提示词设计、检索增强生成等技术, 克服 LLM 过度依赖静态训练数据导致的偏见, 提高生成质量.

此外, 通用 LLM 存在教育适配性的矛盾. 当前大模型生成的回答均倾向于语言通俗化, 强实践性. 这是由于模型训练目标偏向人类偏好, 即相比严谨性更注重流畅可读性. 然而, 尽管这有助于初学者入门, 但过度依赖示例导致理论深度存在一定不足, 且缺乏对知识点的层层递进, 不利于进一步深入学习. 为进一步推进大语言模型在教育

上的深入使用, 有必要在基础模型上进行针对教育的微调.

表 5 大模型对歧义概念的回答

概念和方法名称	回答1	回答2
反模型 (anti-model) 测试	它专注于验证系统在面对不符合预期模型或规则的输入时的行为表现	通过测试探索模型, 再根据探索出来的模型进行基于模型的测试
领域 (domain) 测试	测试软件在特定输入域内的行为	测试软件应用程序在特定业务领域或行业中的功能和行为
嵌入式 (embedded) 测试	将测试人员或测试团队嵌入到开发团队中, 在整个软件开发过程中与开发人员紧密合作, 以便能够尽早发现和解决问题	针对嵌入式系统进行的软件测试活动
备份 (backup) 测试	对系统或数据备份和恢复过程的测试	对已经通过主要测试的功能进行再次验证, 以确保这些功能在其他变更或更新后依然有效
余量 (remainder) 测试	关注于测试那些在主要测试范围之外的剩余功能或场景	余数测试通常涉及对程序中涉及整数除法的逻辑进行验证
极限 (extreme) 测试	一种强调频繁和全面测试的方法	专注于在极端或临界条件下测试应用程序的行为

5) 大模型辅助教材编写的可行路径是什么?

基于上述研究问题的观察, 基于大模型的软件测试概念文本生成为教材编写提供了一套良好辅助工具. 其主要优势在于, 提供了在知识广度, 内容深度, 时效性上更为全面的文本. 结合智能体和一整套自动化脚本, 可以在最小化人工干预的情况下实现持续迭代, 大大降低编写成本. 其中一条可行路径如下.

(i) 内容生成与筛选. 利用多智能体协同结合检索增强生成技术, 批量生成候选概念文本内容. 生成文本通过智能化初筛 (大模型评估) 和领域专家修正, 得到可用文本.

(ii) 知识结构化整合. 将输出内容按知识体系重新组织, 形成逻辑连贯的教材框架.

(iii) 动态更新. 建立修订智能体, 定期扫描权威期刊和会议的最新研究, 自动提示需修订的章节.

这套流程已在《软件测试的概念与方法》的修订中得到了应用. 通过大模型生成补充材料和专家监督, 该书新增了一章智能软件内容, 并在传统方法章节中增加了数个工业界案例, 习题库扩充了约 20% 的开放性题目. 小规模教学反馈显示, 修订版教材在内容新颖性和案例实用性方面得到了高分评价. 本辅助框架也将在进一步重构完善后进行开源. 未来工作将聚焦于构建人机协同的教材编写范式, 进一步优化生成内容的精确性与教育性.

5 有效性威胁

本文提出并实验了一种大模型能力的测试方法, 但目前仅聚焦于软件测试领域的概念与方法, 因此结果的普适性仍需进一步验证. 值得注意的是, 该方法框架本身具有跨领域可迁移性, 如在医学领域可应用于诊断与治疗知识的问答, 在法律领域可用于法条与案例的评估. 未来结合多学科的实验评估, 将能更全面地考察大模型在领域知识问答的可用性.

大语言模型正在发展过程中, 新的大模型在不断涌现, 成熟且具有影响力的大模型也有很多, 本文只选择了 5 个有代表性的 LLM: DeepSeek、Qwen、Claude、o3 和 Llama, 更多的大模型以及后面新的大模型出现后, 可能会对本文的实验产生不同的结果.

软件测试领域的概念和方法有很多, 而且也在不断发展更新. 据最新的整理和统计, 已经超过 100. 本文只考虑其中 100 个概念, 如果采用更多或者更新的概念, 也有可能发现不一样的问题.

本文针对每个概念只提出了 7 个问题, 这 7 个问题之间也有一定的联系, 可以问的问题不止这些, 更多的问题也许会有更多的发现. 限于研究能力和兴趣, 我们只关注了这些问题, 并且认为这些问题已经能够较为全面地覆盖软件测试领域相关知识. 针对每个问题, 我们对 5 个大模型只采取了一种提示下的回答, 更详细的提示或进行多次提问可能会有更好的结果.

对于 LLM 输出的度量和评估, 我们只选择 4 种度量指标, 并采用人工评估方式. 因此, 评估结果可能会因为度

量指标的局限性而存在缺陷. 同时由于人工度量工作量大而繁琐, 有可能造成误差或误判. 但是我们通过多人共同评价给分, 尽可能减少了这种误差.

6 相关工作及比较

随着大语言模型在软件测试等专业领域的广泛应用, 对其进行科学评估变得愈发重要. 虽然现有的基准测试涵盖了推理、安全及对齐等多个维度[34], 但在知识密集型评估方面仍存在显著挑战.

目前, 针对 LLM 知识广度与深度的评估已有一系列研究. LAMA[35]和 KoLA[36]等主流基准主要通过知识补全或简单的三元组 (主语-关系-宾语) 来测试模型, 这虽然能反映事实性与常识性知识, 却难以描述专业领域中错综复杂的技术关系. 即便是针对医学、金融等特定领域的评估, 也多侧重于特定的下游任务或标准化考试[37]. 这种基于标准答案的评估方式能够捕捉知识体系的数个侧面, 但无法充分反映模型对领域知识的综合理解与合成能力.

而在软件工程教育这一具体的知识密集型领域, 大模型的进展引发了广泛讨论, 一方面有人担心学生会利用这些人工智能工具规避学习, 另一方面也有人对它们可能带来的新型学习方式感到兴奋. 然而, 鉴于这些工具尚处于早期阶段, 人们对其在不同教育环境中的实际表现以及对传统教学形式可能带来的潜在机会或风险还缺乏充分认识. Jalil 等人[38]研究了 ChatGPT 在流行的软件测试课程中的潜在适用性[15], 结果表明 ChatGPT 能够回答 77.5% 的问题, 其中仅 55.6% 正确或部分正确, 且在自我判断正确性方面表现不佳, 回答往往与课程需求不符. 这一发现引发了如何检测和规范学生使用 ChatGPT 的关注, 以确保评估能够真实反映其对课程材料的理解. 与此同时, 研究也发现, 虽然学生使用 ChatGPT 规避评估是一个潜在的危险, 但将 ChatGPT 融入课堂也有几个很有前景的方向. ChatGPT 能够为 53.0% 的问题提供正确或部分正确的解释, 使用某些提供额外问题背景的提示策略可以提高正确答案和解释的概率. 这一发现说明在经过设计的课堂活动或实验中, ChatGPT 有潜力作为辅助工具, 替代部分教师或助教的指导工作, 帮助学生更好地理解课程内容. Mezzaro 等人[39]则从实证角度进一步揭示了学生与 AI 助手的交互模式. 他们在“Code Defenders” 平台的软件测试练习中引入类 ChatGPT 工具, 发现学生对智能助手抱有不切实际的期望, 盲目信任它产生的任何输出, 并经常试图直接使用它来获得测试练习的解决方案. 因此, 经常使用智能助手的学生比不使用的学生效率更低. 但他们乐观地认为, 若能正确引导, 此类工具可以大幅提升软件测试学习的趣味性和互动性.

本文针对 LLM 的核心能力, 即对文本数据的阅读理解、回答问题和文本生成能力, 一方面, 通过设计的一组问题来完成 2013 年的软件测试教材《软件测试的概念与方法》[7]的修订任务, 另一方面, 通过这个任务来系统测试 LLM 对领域知识的综合理解与合成能力. 同时, 我们也考虑将来如何利用 LLM 的辅助进行更加生动有趣的教学. 因此, 本研究构成了软件测试与 LLM 之间的双向评估与深入探索: 一方面探索 LLM 赋能软件测试教学与教材编修的潜力, 另一方面则利用该领域的复杂需求系统评估 LLM 的核心能力.

7 结论及未来工作

大语言模型已成为推动人工智能进步的重要力量, 为教育、科研与工程等众多领域带来全新可能. 然而, 随着其在专业领域的广泛应用, LLM 的可信性、可靠性与可用性问题也日益凸显. 为了更深入地理解当前主流大模型在专业领域问答中的真实能力, 本文聚焦软件测试这一关键技术领域, 从教材知识出发, 构建了包含 100 个概念与方法、每项设定 7 个关键问题的评估基准, 并对 5 个主流大模型的回答进行系统化人工评审.

研究结果表明, 当前 LLM 在大多数测试任务中已具备较强的文本生成能力, 尤其在准确性、完整性与语言流畅性方面表现良好, 具备支持专业教材编写、知识获取与教学实践的潜力. 然而, 在部分复杂问题上, 幻觉与推理偏差仍然存在, 尤其是在需要深层逻辑推演或具体定义辨析的任务中表现不稳, 凸显出 LLM 在专业语境理解与知识边界识别方面的局限性.

本研究展示了利用专业领域问答对 LLM 进行可信性评估的新路径, 提出了一套可复用的评估流程, 并构建了一个开源的基准测试集, 为后续开展大模型评测与教育应用研究奠定基础. 未来, 我们将进一步扩展测试覆盖面,

构建更大规模、更具挑战性的问题集合, 以实现对 LLM 能力的更全面衡量.

此外, 构建面向软件测试教育的定制大语言模型也是值得探索的重要方向. 通过限定问题子集、提供任务引导与个性化教学建议等方式, 可打造更加精准、可控与教育友好的智能系统, 助力提升教学质量与学习体验.

综上所述, LLM 在专业领域具备广阔应用前景, 但其性能评估、幻觉控制与教育集成仍面临诸多挑战. 系统化评估不仅是推动大模型可靠发展的关键步骤, 也将成为智能教育、自动化知识服务等新兴领域的基础保障.

References

[1] Ouedraogo WC, Kabore K, Song YW, Klein J, Tian HY, Koyuncu A, Lo D, Bissyande TF. LLMs and prompting for unit test generation: A large-scale evaluation. In: Proc. of the 39th IEEE/ACM Int'l Conf. on Automated Software Engineering. Sacramento: ACM, 2024. 2464–2465. [doi: 10.1145/3691620.3695330]

[2] Guo XJ, Li C, Tsuchiya T. Boundary value test input generation using a large language model: Fault detection and coverage analysis. In: Yoshikawa M, Meng X, Cao Y, Xiao C, Chen W, Wang Y, eds. Advanced Data Mining and Applications. Singapore: Springer, 2025. [doi: 10.1007/978-981-95-3459-3_32]

[3] Widyasari R, Ang JW, Nguyen TG, Sharma N, Lo D. Demystifying faulty code: Step-by-step reasoning for explainable fault localization. In: Proc. of the 2024 IEEE Int'l Conf. on Software Analysis, Evolution and Reengineering (SANER). Rovaniemi: IEEE, 2024. 568–579. [doi: 10.1109/SANER60148.2024.00064]

[4] Abdullin A, Derakhshanfar P, Panichella A. Test wars: A comparative study of SBST, symbolic execution, and LLM-based approaches to unit test generation. In: Proc. of the 2025 IEEE Conf. on Software Testing, Verification and Validation (ICST). Napoli: IEEE, 2025. 221–232. [doi: 10.1109/ICST62969.2025.10989033]

[5] Liu KB, Chen ZP, Liu YY, Zhang JM, Harman M, Han YD, Ma Y, Dong YH, Li G, Huang G. LLM-powered test case generation for detecting bugs in plausible programs. In: Proc. of the 63rd Annual Meeting of the Association for Computational Linguistics. Vienna: ACL, 2025. 430–440. [doi: 10.18653/v1/2025.acl-long.20]

[6] Chowdhury SR, Sridhara G, Raghavan AK, Bose J, Mazumadr S, Singh H, Sugumaran SB, Britto R. Static program analysis guided LLM based unit test generation. In: Proc. of the 8th Int'l Conf. on Data Science and Management of Data. Jodhpur: ACM, 2025. 279–283. [doi: 10.1145/3703323.3703742]

[7] Nie CH. Concepts and Methods of Software Testing. Beijing: Tsinghua University Press, 2013. 205 (in Chinese).

[8] Zhao WX, Zhou K, Li JY, Tang TY, Wang XL, Hou YP, Min YQ, Zhang BC, Zhang JJ, Dong ZC, Du YF, Yang C, Chen YS, Chen ZP, Jiang JH, Ren RY, Li YF, Tang XY, Liu ZK, Liu PY, Nie JY, Wen JR. A survey of large language models. arXiv:2303.18223, 2024.

[9] Song D, Xie X, Song JY, Zhu DR, Huang YH, Juefei-Xu F, Ma L. LUNA: A model-based universal analysis framework for large language models. IEEE Trans. on Software Engineering, 2024, 50(7): 1921–1948. [doi: 10.1109/TSE.2024.3411928]

[10] Chang YP, Wang X, Wang JD, Wu Y, Yang LY, Zhu KJ, Chen H, Yi XY, Wang CX, Wang YD, Ye W, Zhang Y, Chang Y, Yu PS, Yang Q, Xie X. A survey on evaluation of large language models. ACM Trans. on Intelligent Systems and Technology, 2024, 15(3): 39. [doi: 10.1145/3641289]

[11] Wang JJ, Huang YC, Chen CY, Liu Z, Wang S, Wang Q. Software testing with large language models: Survey, landscape, and vision. IEEE Trans. on Software Engineering, 2024, 50(4): 911–936. [doi: 10.1109/TSE.2024.3368208]

[12] Yang L, Yang C, Gao ST, Wang WJ, Wang B, Zhu QH, Chu X, Zhou JY, Liang GT, Wang QX, Chen JJ. On the evaluation of large language models in unit test generation. In: Proc. of the 39th IEEE/ACM Int'l Conf. on Automated Software Engineering (ASE 2024). Sacramento: ACM, 2024. [doi: 10.1145/3691620.3695529]

[13] Schäfer M, Nadi S, Eghbali A, Tip F. An empirical evaluation of using large language models for automated unit test generation. IEEE Trans. on Software Engineering, 2024, 50(1): 85–105. [doi: 10.1109/TSE.2023.3334955]

[14] Alshahwa N, Chheda J, Finogenova A, Gokkaya B, Harman M, Harper I, Marginean A, Sengupta S, Wang E. Automated unit test improvement using large language models at Meta. In: Proc. of the 32nd ACM Int'l Conf. on the Foundations of Software Engineering (Compaion Volume). Porto de Galinhas: ACM, 2024. 185–196. [doi: 10.1145/3663529.3663839]

[15] Molina F, Gorla A, D'Amorim M. Test oracle automation in the era of LLMs. ACM Trans. on Software Engineering and Methodology, 2025, 34(5): 150. [doi: 10.1145/3715107]

[16] Xue ZY, Li LG, Tian SY, Chen XH, Li PP, Chen LY, Jiang TT, Zhang M. LLM4Fin: Fully automating LLM-powered test case generation for FinTech software acceptance testing. In: Proc. of the 33rd ACM SIGSOFT Int'l Symp. on Software Testing and Analysis (ISSTA 2024). Vienna: ACM, 2024. 1643–1655. [doi: 10.1145/3650212.3680388]

[17] Augusto C, Morán J, Bertolino A, de La Riva C, Tuya J. Software system testing assisted by large language models: An exploratory

study. In: Proc. of the 36th IFIP WG 6.1 Int'l Conf. on Testing Software and Systems. 2025. 239–255. [doi: 10.1007/978-3-031-80889-0_17]

[18] Li YH, Liu P, Wang HY, Chu J, Wong WE. Evaluating large language models for software testing. Computer Standards & Interfaces, 2025, 93: 103942. [doi: 10.1016/j.csi.2024.103942]

[19] Deljouyi A, Koohestani R, Izadi M, Zaidman A. Leveraging large language models for enhancing the understandability of generated unit tests. In: Proc. of the 47th IEEE/ACM Int'l Conf. on Software Engineering (ICSE). Ottawa: IEEE, 2025. 1449–1461. [doi: 10.1109/ICSE55347.2025.00032]

[20] Hossain SB, Dwyer MB. TOGLL: Correct and strong test oracle generation with LLMS. In: Proc. of the 47th IEEE/ACM Int'l Conf. on Software Engineering (ICSE). Ottawa: IEEE, 2025. 1475–1487. [doi: 10.1109/ICSE55347.2025.00098]

[21] Cheng X, Sang F, Zhai YZ, Zhang XK, Kim T. Rug: Turbo LLM for Rust unit test generation. In: Proc. of the 47th IEEE/ACM Int'l Conf. on Software Engineering (ICSE). Ottawa: IEEE, 2025. 2983–2995. [doi: 10.1109/ICSE55347.2025.00097]

[22] Li JG, Dong Z, Wang C, You HZ, Zhang C, Liu Y, Peng X. LLM based input space partitioning testing for library APIs. In: Proc. of the 47th IEEE/ACM Int'l Conf. on Software Engineering (ICSE). Ottawa: IEEE, 2025. 1436–1448. [doi: 10.1109/ICSE55347.2025.00153]

[23] Li TO, Zong WX, Wang YB, Tian HY, Wang Y, Cheung CS, Kramer J. Nuances are the key: Unlocking ChatGPT to find failure-inducing tests with differential prompting. In: Proc. of the 38th IEEE/ACM Int'l Conf. on Automated Software Engineering (ASE 2023). Luxembourg: IEEE, 2023. 14–26. [doi: 10.1109/ASE56229.2023.00089]

[24] Hendrycks D, Burns C, Basart S, Zou A, Mazeika M, Song D, Steinhardt J. Measuring massive multitask language understanding. In: Proc. of the 9th Int'l Conf. on Learning Representations (ICLR). OpenReview.net, 2021.

[25] Srivastava A, Rastogi A, Rao A, *et al*. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. arXiv:2206.04615, 2022.

[26] Wang A, Pruksachatkun Y, Nangia N, Singh A, Michael J, Hill F, Levy O, Bowman SR. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In: Proc. of the 33rd Conf. on Neural Information Processing Systems (NeurIPS 2019). Vancouver: NeurIPS, 2019. 3261–3275.

[27] Jin Q, Dhingra B, Liu ZP, Cohen WW, Lu XH. PubMedQA: A dataset for biomedical research question answering. In: Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int'l Joint Conf. on Natural Language Processing (EMNLP-IJCNLP 2019). Hong Kong: ACL, 2019. 2567–2577. [doi: 10.18653/v1/D19-1259]

[28] Holzenberger N, Blair-Stanek A, van Durme B. A dataset for statutory reasoning in tax law entailment and question answering. In: Proc. of the 24th Natural Legal Language Processing Workshop. San Diego: NLLP, 2020. 31–38.

[29] Chen M, Tworek J, Jun H, *et al*. Evaluating large language models trained on code. arXiv:2107.03374, 2021.

[30] Perez E, Huang S, Song F, Cai T, Ring R, Aslanides J, Glaese A, McAleese N, Irving G. Red teaming language models with language models. In: Proc. of the 2022 Conf. on Empirical Methods in Natural Language Processing. Abu Dhabi: ACL, 2022. 3419–3448. [doi: 10.18653/v1/2022.emnlp-main.225]

[31] Liang P, Bommasani R, Lee T, *et al*. Holistic evaluation of language models. arXiv:2211.09110, 2022.

[32] Ji ZW, Lee N, Frieske R, Yu TZ, Su D, Xu Y, Ishii E, Bang YJ, Madotto A, Fung P. Survey of hallucination in natural language generation. ACM Computing Surveys, 2023, 55(12): 248. [doi: 10.1145/3571730]

[33] Biderman S, Schoelkopf H, Sutawika L, *et al*. Lessons from the trenches on reproducible evaluation of language models. arXiv:2405.14782, 2024.

[34] Guo ZS, Jin RR, Liu C, Huang YF, Shi D, Supryadi, Yu LH, Liu Y, Li JX, Xiong BJ, Xiong DY. Evaluating large language models: A comprehensive survey. arXiv:2310.19736, 2023.

[35] Petroni F, Rocktäschel T, Riedel S, Lewis P, Bakhtin A, Wu YX, Miller A. Language models as knowledge bases? In: Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int'l Joint Conf. on Natural Language Processing. Hong Kong: ACL, 2019. 2463–2473. [doi: 10.18653/v1/D19-1250]

[36] Yu JF, Wang XZ, Tu SQ, *et al*. KoLA: Carefully benchmarking world knowledge of large language models. In: Proc. of the 12th Int'l Conf. on Learning Representations. Vienna: ICLR, 2024.

[37] Antaki F, Touma S, Milad D, El-Khoury J, Duval R. Evaluating the performance of ChatGPT in ophthalmology: An analysis of its successes and shortcomings. Ophthalmology Science, 2023, 3(4): 100324. [doi: 10.1016/j.xops.2023.100324]

[38] Jalil S, Rafi S, LaToza TD, Moran K, Lam W. ChatGPT and software testing education: Promises & perils. In: Proc. of the 2023 IEEE Int'l Conf. on Software Testing, Verification and Validation Workshops (ICSTW). Dublin: IEEE, 2023. 4130–4137. [doi: 10.1109/ICSTW58534.2023.00078]

[39] Mezzaro S, Gambi A, Fraser G. An empirical study on how large language models impact software testing learning. In: Proc. of the 28th

Int'l Conf. on Evaluation and Assessment in Software Engineering (EASE 2024). Salerno: ACM, 2024. 555–564. [doi: 10.1145/3661167.3661273]

附中文参考文献

[7] 聂长海. 软件测试的概念与方法. 北京: 清华大学出版社, 2013. 205.

作者简介

陈煜磊, 博士生, CCF 学生会员, 主要研究领域为智能化软件测试, LLM 测试.

聂钰格, 博士生, CCF 学生会员, 主要研究领域为移动软件测试, LLM 测试.

吴化尧, 博士, 助理教授, CCF 专业会员, 主要研究领域为自动化和智能化软件测试.