大模型背景下软件测试课程理解验证型教学模式构建与实践

邹燕飞

咸阳师范学院 计算机学院，中国·陕西 咸阳 712000

摘要：大语言模型进入软件测试课堂后，学生获得测试代码的速度明显加快，但“会生成”并不等同于“会测试”。针对课程实施中出现的 AI 依赖、边界分析不足和结果性评价失真等现象，本文结合《软件测试技术与实践》课程，构建“AI 协作—项目驱动—理解验证”教学模式。研究以软件工程专业大三 2 个教学班 76 名学生为对象，围绕 pytest/unittest、Selenium、Postman 3 类实践模块，设置 Bug 狩猎、边界挑战、差异化被测对象、断言逐行说明等任务，并将阶段记录、现场演示、口头追问和项目报告纳入评价。实践结果显示，学生对该教学模式总体比较认可，其中认可“能检验真实掌握情况”和“有助于理解接口测试逻辑”的比例分别为 94.44% 和 91.67%；表现性评价平均分为 24.42/30。分项结果中，关键代码或断言解释、随机追问应答得分率低于现场运行与流程说明，提示该模式能够暴露“脚本可运行”与“逻辑可解释”之间的差距，可为大模型背景下编程实践类课程的任务重构和评价改进提供参考。

关键词：大语言模型；软件测试；AI 协作；项目式学习；理解验证

Construction and Practice of an Understanding–Verification–Oriented Teaching Model for Software Testing Courses in the Context of Large Language Models

Zou Yanfei

School of Computer Science, Xianyang Normal University, China Shaanxi Xianyang 712000

Abstract: The introduction of LLM into software testing courses has substantially increased the speed at which students can produce test code. However, being able to generate test scripts does not necessarily mean being able to conduct effective testing. To address problems observed in course implementation, including overreliance on AI, insufficient attention to boundary conditions, and distortions in outcome-based assessment, this study developed an AI-assisted, project-driven, and understanding-oriented verification teaching model for the course Software Testing Technology and Practice. The study involved 76 third-year Software Engineering students from two classes. Three practical modules—pytest/unittest, Selenium, and Postman—were redesigned to include activities such as bug hunting, boundary-testing challenges, differentiated systems under test, and line-by-line explanation of assertions. Process records, live demonstrations, oral questioning, and project reports were incorporated into the assessment. The results showed that students generally responded positively to the teaching model. In particular, 94.44% agreed that it was effective in assessing their actual level of mastery, while 91.67% reported that it helped them better understand the logic of API testing. The mean score for performance-based assessment was 24.42 out of 30. Further analysis showed that students performed less well in explaining key code or assertions and responding to spontaneous questions than in executing tests and describing testing procedures. These findings suggest that the proposed model can reveal the gap between making a test script run and being able to explain the testing logic behind it. The study provides a practical reference for redesigning learning tasks and assessment approaches in programming-oriented courses in the context of large language models.

Keywords: LLM; Software testing; AI collaboration; Project-based learning; Understanding verification

0 引言

以 DeepSeek、ChatGPT 为代表的大语言模型，已经成为学生完成编程实践任务时经常调用的工具。它们可以在较短时间内给出 pytest 测试函数、Selenium 自动化脚本或 Postman 断言示例，降低了学生进入任务的技术门槛。与此同时，已有研究也注意到，AI 代码生成工具会改变编

程课程的任务设计、学术诚信判断和学习评价方式[1-4]；计算机教育领域关于知识图谱、师生机协同和过程性评价的研究，则为软件类课程改革提供了可借鉴的思路[5-7]。在软件测试课程中，这一变化尤其明显：教师看到一份结构完整的测试文件时，并不能据此确认学生是否理解了测试目标、边界条件和断言依据。

软件测试课程的教学重点并不只是让学生写出一段可以运行的代码，而是训练其判断软件行为是否符合需求的能力。缺陷发现、等价类划分、边界值分析、断言设计和结果解释，构成了课程中更关键的认知活动。大模型可以补全语法和框架，却难以代替学生理解业务规则、识别异常分支、判断测试是否充分。因此，在大模型已经进入学习过程的背景下，课程改革的重点不宜停留在“禁用”或“放任”AI 的二元选择上，而应重新设计任务和评价，使 AI 的作用被限定在支架层面，学生的理解过程仍然可观察、可追问、可评价。

1 研究设计与教学模式构建

1.1 研究对象与数据来源

本研究依托《软件测试技术与实践》课程开展相关教学实践探索。研究选取软件工程专业大三学生作为调研样本，共涵盖两个平行教学班 76 名学生。作为专业核心必修课，该课程总学时设置为 32，理论教学与实践教学各占 16 学时。课程教学内容主要分为三个模块，分别是 pytest/unittest 单元测试、Selenium 自动化测试与 Postman 接口测试，整个教学实施周期横跨 16 教学周。课程结束后共回收有效问卷 72 份，有效回收率达 94.74%；研究同时选取 12 名学生开展半结构化访谈，受访学生的前期学习基础、课程项目完成进度存在明显差异，另外还对 2 名任课教师及教学辅助人员进行了访谈。数据来源如表 1 所示。

本次研究的数据收集环节同时覆盖学习过程与学习结果两类维度。过程性采集材料包含课堂观察记录、在线学习平台行为日志、阶段性学习检查材料以及学生自主提交的 AI 工具使用说明，这类材料主要用于核验学生是否完整经历需求理解、用例设计、脚本调试与错误修正的完整测试流程；结果性采集材料则包含期末课程展示答辩记录、问卷反馈内容与访谈转录文本，主要用于分析学生对本次教学模式的接纳程度，以及其测试逻辑的书面与口头表达水平。需要说明的是，本轮教学实践未设置严格的随机对照班级，因此本研究的定位为教学模式构建与探索性实践研究，核心呈现内容为模式设计逻辑、具体实施路径与初步应用效果。

表 1 数据来源及用途

数据来源	观察重点	用途
问卷调查	AI 使用、任务感知、评价认可度	了解学生整体反馈
课堂观察	参与行为、提问讨论、AI 调用方式	识别学习过程变化
平台记录	签到、提交、讨论、阶段检查	反映持续投入情况
期末答辩	现场运行、逻辑解释、随机追问	检验真实理解水平
学生访谈	使用动机、困难、学习收获	解释模式作用机制

1.2 模式框架

本教学模式由“AI 协作—项目驱动—理解验证”三个核心环节搭建完成，整体框架呈现可参考图 1。AI 协作环节主要回应学生初入实践任务时的起步障碍问题，学生可借助提示词调试、示例代码参考、报错原因分析三类路径获取初始学习支架。项目驱动环节突出任务与具体业务场

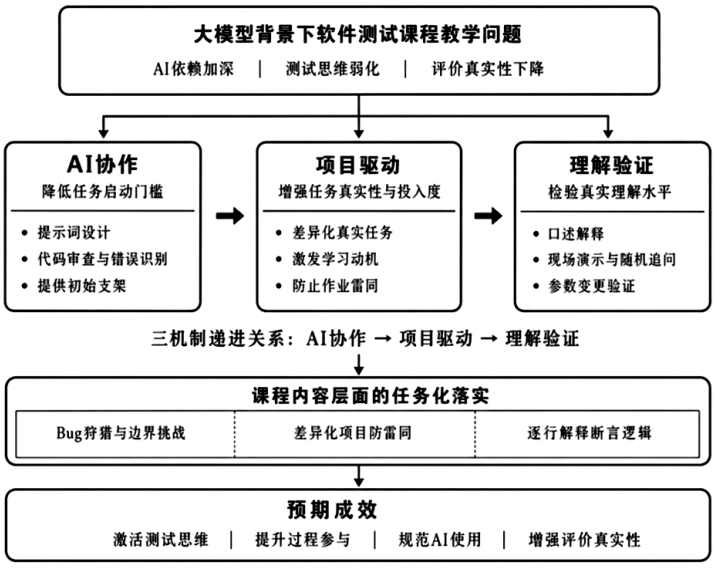

图 1“AI 协作—项目驱动—理解验证”教学模式框架

景的绑定要求，依托差异化被测对象、多元化业务流程设置，压缩标准答案式作业的存在空间。理解验证环节将口述逻辑说明、现场程序运行、随机问题追问、参数调整测试四类表现纳入评价维度，据此判断学生对测试逻辑的实际掌握程度。

AI 协作并不意味着允许学生将作业任务整体外包给大模型，而是要将模型的使用边界严格限定在可审查的辅助定位上。项目驱动也不是简单在课程末尾增设一次综合性大作业，而是通过千人千面的个性化任务设置，抬高直接复制现成答案的违规成本。理解验证更不是期末阶段临时增设的答辩环节，而是贯穿选题、开发、调试、成果评价全流程的能力核验机制。缺少 AI 协作的支撑，基础薄弱的学生很难跨过复杂任务的入门门槛；去掉项目驱动的场景约束，AI 工具极容易异化为学生直接获取标准答案的捷径；缺失理解验证的过程性把关，最终提交的项目成果也无法真实反映学生的实际能力掌握水平。该模式的长期实施效果还有待更大样本的教学数据验证。

这三者并非孤立存在，而是形成了逐层推进的内在关联。AI 协作的核心作用是把学生牵引到具体测试任务场景中，项目驱动的设计思路又能牢牢锚定 AI 的辅助定位，避免工具越位替代学生完成核心学习过程。理解验证环节的设置，主要目的是保障最终学习成果可落地转化为可观测、可评估的实际测试能力。

这套运行模式划定了 AI 参与软件测试课程教学活动的适用边界。AI 可承接需求拆解解读、基础代码框架生成类工作，也可辅助完成测试数据构造、程序错误原因排查等事务性任务。需要说明的是，测试目标的自主判定、边界条件的独立识别、断言背后业务逻辑的阐释这几类核心思维训练内容，必须由学生独立完成，不可交由 AI 代劳。任课教师开展教学活动时，并未将学生使用 AI 的行为判定为违规，反而将“能否独立审查、甄别 AI 输出内容的合理性”纳入测试能力培养的核心范畴。教师给学生划定的使用规则为：学生可以借助 AI 完成相关任务，但必须逐一明确说明 AI 具体承担了哪部分工作、个人对 AI 输出内容做了哪些调整修改、做出对应调整的具体依据是什么。

1.3 实施流程

整个教学环节按“方向探索—框架搭建—迭代开发—展示答辩”的路径逐层推进，每个阶段都提前划定学生、AI 工具、教师三方的职责边界。方向探索环节由学生自主提出感兴趣的测试对象，AI 可协助完成需求的初步梳理，教师的核心工作是判断任务规模是否适配教学周期、方案是否具备落地可行性；框架搭建阶段，AI 可输出基础代码片段或示例结构供参考，学生需要独立理解代码逻辑并完成原型跑通，教师负责整体技术路线的合理性把控；迭代开发环节，学生自主补充测试用例、完成脚本调试并同步记录每一处修改过程，教师对照预设检查点给出针对性调整反馈；进入展示答辩阶段，学生需脱离 AI 的即时辅助，独立完成现场项目演示与评委追问应答。

对学生理解程度的验证并非只集中在最终答辩环节，而是贯穿嵌入四个教学阶段的全过程。方向探索阶段，学生需要明确阐述选定该测试对象的原因，梳理清楚测试覆盖的核心业务流程；框架搭建阶段，学生要能清晰解释测试框架的选型逻辑，说明初始代码结构的设计思路；迭代开发阶段，学生需要提交完整的修改记录、错误定位的完整思考过程，以及每一项新增测试用例的设计依据；展示答辩阶段，学生需要现场完成项目运行，回应教师围绕断言逻辑、边界条件设置、参数变化影响提出的各类问题。这种贯穿全程的连续性检查，能帮助教师尽早发现“代码完整但学生对核心逻辑理解不足”的问题。

2 差异化任务与评价设计

2.1 三类测试模块任务设计

结合课程能力培养目标，教学模式分别应用于 pytest/unittest 单元测试、Selenium 自动化测试和 Postman 接口测试三个实践模块，重点训练边界值分析、测试流程设计和接口业务规则校验能力。考核不以提交代码为终点，学生还需说明测试用例的设计依据，审查 AI 生成内容，并通过现场操作和答辩验证真实掌握情况。

单元测试模块采用含有预设缺陷的 AI 生成代码，要求学生定位错误断言，并依据等价类划分和边界值分析补充测试用例。例如，登录功能除正常路径外，还应覆盖空用户名、超长密码、特殊字符和连续输错触发锁定等场景。该任务旨在引导学生区分代码结构完整与测试覆盖充分，将关注点由代码形式转向测试逻辑。Selenium 模块允许学生自主选择公开网站，围绕搜索、登录、表单提交等流程设计自动化测试。由于不同网站的 DOM 结构和交互逻辑存在差异，学生需要处理动态加载、页面延迟、弹窗遮挡和定位失效等问题。作业包括被测对象说明、测试流程图和元素定位表，答辩时需解释定位方式、等待策略及流程控制逻辑，使教学目标由“脚本能够运行”转向“测试过程能够说明”。Postman 模块要求学生针对 Flask 接口设计参数、状态码和返回字段断言，并说明其业务含义。教师通过现场调整参数规则、账号权限和返回字段，要求学生

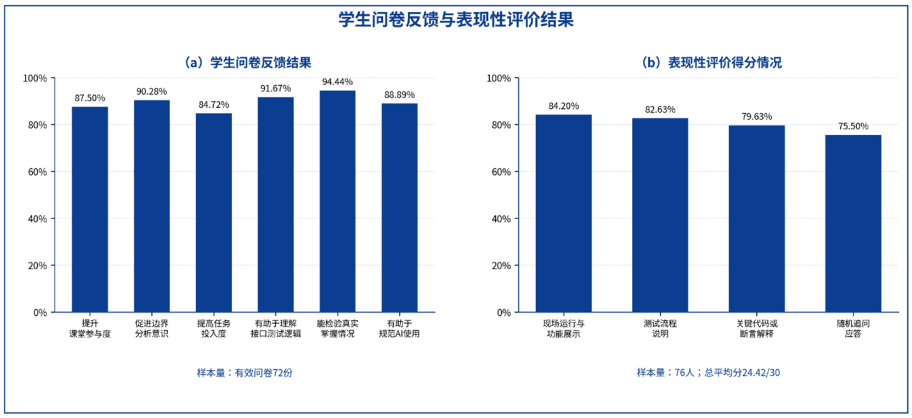

图 2 学生问卷反馈与表现性评价结果

判断预期结果的变化。例如，新增用户接口不仅要验证用户标识，还应覆盖用户名重复、角色非法和必填参数缺失等异常场景。该模块重点考查学生对业务规则、异常分支和漏测风险的理解。

2.2 多元评价体系

为规避“AI 生成作业、AI 形式化批改”引发的评价结果偏差，本课程搭建了融合过程性、表现性、结果性三类维度的多元评价框架，各维度的权重设置与具体观测点可查阅表 2。过程性评价的观测范围覆盖课堂互动参与频次、分阶段任务推进节奏、学习全流程的 AI 工具调用记录三类内容，占评价总权重的 40%。表现性评价环节不依赖提交的静态材料，教师通过现场操作演示、核心思路口述阐释、知识点随机追问三类方式，核验学生对课程内容的真实掌握水平，可有效过滤代笔、AI 代写带来的虚假掌握信号。结果性评价的判定依据包含最终项目完成质量、测试方案的逻辑完整性、课程报告的格式与内容规范性，其评价结果会和前两类评价的得分做交叉校验。

表 2 多元评价体系维度与占比

评价维度	占比	评价内容	评价方式
过程性评价	50%	课堂参与表现、阶段任务推进、AI 使用行为记录、过程性文档	学习平台行为数据、课堂观察记录、阶段性检查点
表现性评价	30%	现场演示、口头答辩、关键代码与断言逻辑解释、随机追问应答	教师围绕核心逻辑追问并现场评分
结果性评价	20%	项目完成度、测试方案完整性、测试报告规范性	项目最终提交文件及测试报告评分

3 教学实践成效

教学实践结束后，共回收有效问卷 72 份，结果见图 2（a）。六个题项的认可度均超过 80%，其中“能检验真实掌握情况”最高，为 94.44%；“有助于理解接口测试逻辑”为 91.67%；“提高任务投入度”相对较低，为 84.72%。访谈显示，差异化项目和现场追问增加了学习压力，但也促使学生由关注代码能否运行，转向分析断言依据、异常输入和业务边界。部分学生希望增加示例和分层指导。期末表现性评价共 76 人参加，平均得分 24.42 分（满分 30），整体得分率为 81.40%，结果见图 2（b）。现场运行与功能展示得分率最高，为 84.20%；随机追问应答最低，为 75.50%。说明学生已基本掌握测试脚本运行和流程说明，但在断言解释、参数变化分析及异常分支判断方面仍需加强。

4 结语

本研究搭建“AI 协作—项目驱动—理解验证”的教学框架中，AI 工具承担两类：一类是学生编写测试代码时的辅助工具，可用于基础代码生成、常见 bug 定位调试；另一类则是测试审查的核心对象，学生需要对 AI 输出的测试代码做校验。教学实施阶段会根据学生的能力层级布置差异化的项目实践任务，同时结合口述原理解释、操作演示、随机知识点追问、测试参数调整等方式，夯实理解验证环节的考核权重。学期末的课程实践数据显示，学生逐步开始主动关注测试设计依据、断言逻辑合理性的占比提升，学生借助 AI 工具完成作业的行为得到了一定规范，课程评价结果和学生实际能力的匹配度也出现明显改善。后续教学过程中，会持续扩大实践样本的覆盖范围，搭建 AI 助教的作业预审机制。

参考文献：

[1] Becker B A, Denny P, Finnie-Ansley J, et al. Programming is hard—or at least it used to be: educational opportunities and challenges of AI code generation[C]// Proceedings of the 54th ACM Technical Symposium on Computer Science Education. New York: ACM, 2023:500-506. DOI:10.1145/3545945.3569759.

[2] Kasneci E, Sessler K, K ü chemann S, et al. ChatGPT for good? On opportunities and challenges of large language models for education[J]. Learning and Individual Differences, 2023, 103:102274. DOI:10.1016/j.lindif.2023.102274.

[3] Prather J, Denny P, Leinonen J, et al. The robots are here: navigating the generative AI revolution in computing education[C]//Proceedings of the 2023 Working Group Reports on Innovation and Technology in Computer Science Education. New York: ACM, 2023:108-159. DOI:10.1145/3623762.3633499.

[4] Denny P, Kumar V, Giacaman N. Conversing with Copilot: exploring prompt engineering for solving CS1 problems using natural language[C]//Proceedings of the 54th ACM Technical Symposium on Computer Science Education. New York: ACM, 2023:1136-1142. DOI:10.1145/3545945.3569823.

[5] 王云艳. 知识图谱驱动的 C++ 课程 BOPPPS+ ADCD 教学模式探索[J/OL]. 计算机教育，2026(1):233-239. DOI:10.16512/j.cnki.jsjjy.2026.01.021.

[6] 吴青，程筱晏，赵广辉等. AIGC 辅助计算机编程教学的机遇和挑战[J/OL]. 计算机教育，2025(7):69-73. DOI:10.16512/j.cnki.jsjjy.2025.07.022.

[7] 杨华利，阮晓莉，王晨等. 新工科背景下数据驱动的师生机协同的计算思维培养[J/OL]. 计算机教育，2026(1):42-48. DOI:10.16512/j.cnki.jsjjy.2026.01.036.

基金项目：咸阳师范学院 2025 年校级教项目 人工智能专项，[2025ZXY17] 大模型时代计算机编程课程教学改革与实践探索。