大模型时代计算机科学与技术专业教学范式重构研究

柯文俊 汪鹏 吕美香

东南大学 计算机科学与工程学院，南京 211189

摘 要 针对大模型进入计算机专业课堂后作业真实性弱化、评价效度下降和基础能力空心化问题，本文结合文献与调查数据，分析代码生成对学习与评价的影响。数据显示，英国本科生 AI 使用率由 2024 年的 66%升至 2025 年的 92%，2026 年已有 95%的学生使用 AI，94%用于被评分任务。基于此，构建“基础能力—AI 协同—输出审查—系统责任”四层能力模型，提出无 AI 基础考核与 AI 协同项目并行的双轨评价，设计问卷、准实验、学习日志和访谈组成的证据链。研究认为，大模型可提高任务效率，但不必然提升概念理解与独立编程能力；教学应将问题建模、输出验证和工程责任纳入课程目标，推动评价转向过程证据、AI 使用转向规范协同。

关键词 大模型；计算机科学教育；编程教学；生成式人工智能；教学评价；人机协同

Reconstructing the Teaching Paradigm of Computer Science and Technology in the Era of Large Language Models

Wenjun Ke Peng Wang Meixiang Lv

School of Computer Science and Engineering Southeast University

Nanjing 211189, China

kewenjun@seu.edu.cn pwang@seu.edu.cn meixianglv@seu.edu.cn

Abstract—Large language models (LLMs) challenge assignment authenticity, assessment validity, and foundational competence in computer science education. Drawing on prior studies and public survey data, this paper analyzes how code generation reshapes learning and assessment. UK undergraduate AI use rose from 66% in 2024 to 92% in 2025; by 2026, 95% of students used AI and 94% used it for assessed work. We therefore develop a four-layer competency model covering foundational competence, AI collaboration, output auditing, and system responsibility; propose dual-track assessment combining AI-free foundational tests with AI-assisted projects; and design an evidence chain integrating questionnaires, quasi-experiments, learning logs, and interviews. The analysis indicates that LLMs can improve task efficiency without necessarily improving conceptual understanding or independent programming. Computer science programs should incorporate problem modeling, output verification, and engineering responsibility into learning objectives, shifting assessment toward process evidence and AI use toward regulated collaboration.

Keywords—large language models; computer science education; programming education; generative artificial intelligence; instructional assessment; human–AI collaboration

1 引 言

2023 年以来，以 ChatGPT、GitHubCopilot、Claude、Gemini、DeepSeek、通义千问等为代表的大模型工具迅速进入高等教育场景。对于计算机科学与技术专业而言，这种影响尤为直接：代码生成、程序解释、错误定位、单元测试、文档撰写和项目重构等任务，正是大模型最容易形成可见效果的环节。Raihan 等对 125 篇计算机教育相关研究的系统综述表明，大模型在本科编程课程、代码理解、自动反馈和教师辅助方面已形成密集研究热点[1]。

然而，大模型带来的并不只是“效率工具”的更新。传统计算机专业教学长期把学生独立写出代码作为训练核心和评价依据，默认代码结果与学生能力之间存在较强对应关系。但在大模型环境下，一段能够运行的代码可能来自学生理解后的实现，也可能来自对 AI 输出的直接复制。由此，传统编程作业的真实性、过程可解释性和评价效度均受到挑战。Denny 等提出“Prompt Problems”这一新型编程练习，正是因为生成式 AI 时代的编程教育正在从单纯写代码转向提示、阅读、理解和评价 AI 生成代码[2]。

作者结合多年的教学经验和对上述挑战的分析，给出一个基本判断：大模型不是计算机教育的外部辅助工具，而是正在改变计算机知识生产方式的新型学习环境。计算机专业教学的目标不应变成培养“比 AI 更快写代码的人”，而应转向培养能够提出正确问题、拆解复杂任务、验证智能输出、组织工程系统并承担技术责任的人。基于这一判断，本文提出面向大模型时代的计算机专业教学重构框架，并给出可操作的调

*通讯作者：柯文俊 kewenjun@seu.edu.cn

研与课堂准实验方案。国内相关研究也指出，AI 大模型正在对高校计算机教育的人才培养目标、教学内容、教师角色和学习方式产生系统性影响[3]。

2 大模型对计算机专业教学的冲击

2.1 代码产出从稀缺资源转变为低成本资源

在传统教学中，学生写出可运行程序本身具有较高教育价值。因为代码产出的过程包含了需求理解、语法选择、算法组织、调试修改等环节。大模型出现后，这一逻辑发生变化：学生可以通过自然语言描述快速得到可运行代码，甚至可以让模型进一步解释、重构和补充测试。Raihan 等关于大模型完成入门编程作业能力的研究显示，现代模型已能在相当程度上处理典型 CS1 任务[4]。这意味着“代码是否正确”不再足以作为能力判断的唯一依据。

代码不再稀缺，并不意味着编程能力不再重要。恰恰相反，基础编程能力成为判断 AI 输出、发现隐藏错误和进行工程取舍的前提。没有语法、数据结构、算法复杂度和系统边界的基本训练，学生很容易把模型生成的表面正确结果误认为可靠答案。

2.2 作业真实性与评价效度受到挑战

大模型削弱了传统作业中“提交结果—学生能力”的线性关系。HEPI/Kortext 2025 年调查显示，使用 AI 工具的英国本科生比例由 2024 年的 66%上升至 2025 年的 92%，在考核任务中使用生成式 AI 的比例达到 88%[5]。2026 年调查进一步显示，95%的学生至少以某种方式使用 AI，94%的学生使用生成式 AI 辅助被评分任务[6]。这些数据表明，评价制度已经不能建立在“学生很少使用 AI”的假设之上。对于计算机专业，评价效度的下降表现得更为突出。传统机试、实验报告和课程设计主要评价程序能否运行、结果是否正确、报告是否完整。但在大模型辅助下，这些结果性证据无法充分说明学生是否真正理解算法、是否能够独立调试、是否知道代码的安全边界，也无法判断学生在多大程度上进行了人工审查。

2.3 差异由“会写代码”转向“会审查智能”

Lepp 和 Kaimre 对 231 名面向对象程序设计课程学生的研究发现，学生主要将 AI 聊天机器人用于编程任务，但 AI 使用频率与课程表现之间呈负相关[7]。Jošt 等在 React 课程的十周研究中也发现，在代码生成和调试等高认知负荷任务中过度依赖 LLM，可能与学生在无 AI 条件下的最终表现下降相关[8]。这些研究共同提示：AI 使用本身不是学习效果的保证，关键在于学生是否以理解、追问、比较和审查的方式使用 AI。

因此，大模型时代学生之间的差距不再只是“谁写代码更快”，而是“谁更会提出问题、谁更能识别 AI 错误、谁更能设计测试、谁更能解释技术决策”。这也要求计算机专业教学把提示词质量、交互过程、验证策略和输出审查能力纳入课程目标与评价标准。

3 相关工作

3.1 计算机教育中的生成式 AI 研究进展

近两年国际计算机教育研究已经从“AI 能否写代码”转向“如何把 AI 纳入教学设计”。Liu 等在 CS50 课程中构建 AI 辅助工具，使其承担代码解释、风格建议和课程问答功能，并强调通过“教学护栏”避免直接替学生完成任务[9]。其后续研究进一步引入人类反馈与持续评价机制，以保证 AI 助教的回答方式与教学目标一致[10]。这说明，大模型若要进入课程，不能只关注模型能力，还必须关注教学价值对齐和教师可控性。国内研究也已将 ChatGPT 引入程序设计翻转课堂，在课前学习、课堂教学和编程实践等环节探索大模型辅助教学方法[11]。

Xue 等在 CS1 课程实验中发现，ChatGPT 并未显著提升学生在入门编程作业中的学习表现，同时学生在使用 ChatGPT 后对传统学习资源的探索明显减少[12]。Gambo 等 2026 年的混合方法研究也显示，学生对 ChatGPT 的感知收益与实际课程成绩提升之间并不存在直接对应关系，且教师观察到过度依赖和学术诚信风险[13]。这些证据支持本文的核心判断：大模型能够提升完成任务的速度和信心，但不必然提升概念理解和迁移能力。

3.2 提示词、认知参与与学习深度

Mutanga 等对 140 名软件开发学生的 842 条提示词进行分析，发现多数提示停留在记忆、理解和直接求答案层面，高阶的调试、反思和评价型提示比例较低[14]。Ding 等 2026 年的研究进一步提出，学习者与大模型互动不应止于提示工程，而应包含主动发起问题、持续互动和批判性评价，即 HIIC 模型中的 Initiate、Interact 和 Critique[15]。这些研究为本文把“AI 输出审查能力”作为核心能力提供了理论支持。

从计算机专业课程看，提示词不是简单的语言表达技巧，而是问题建模能力、需求澄清能力和工程约束表达能力的外化。一个高质量提示往往包含输入输出约束、算法复杂度要求、边界条件、测试样例和代码风格要求；一个高质量追问则体现学生对模型输出的怀疑、比较和验证。已有程序设计教学实践也将提示工程用于需求表达、代码生成、结果验证和错误修正等教学环节[16]。

3.3 课程标准更新与能力结构变化

ACM、IEEE-CS 与 AAAI 联合发布的 CS2023 课程指南强调，本科计算机教育应从单一知识点覆盖转

向知识、技能和素养结合的能力模型，并更加系统地纳入人工智能、伦理、安全和社会影响等内容[17]。Eaton 和 Epstein 对 CS2023 中人工智能课程内容的阐释也指出，AI 已经不再只是专业方向课程，而是影响本科计算机教育整体设计的重要知识领域[18]。

因此，本文重点不是讨论是否允许学生使用大模型，而是讨论在大模型已经成为学习环境组成部分的条件下，计算机专业应如何重新定义基础训练、高阶能力和评价证据。人工智能技术的发展也要求高校同步调整人才培养目标、课程内容、教学方法和评价机制，以适应智能化环境下能力结构的变化[19]。

4 计算机专业能力模型与教学对比

4.1 “学生基础能力—AI 协同设计”四层模型

第一层是无 AI 基础能力。学生必须在不依赖大模型的条件下掌握基本语法、数据结构、算法思想、复杂度分析、调试方法、计算机组成和操作系统基础。该层能力是判断 AI 输出是否可信的前提，也是防止“能力空心化”的底线。

第二层是 AI 协同编程能力。学生应学会把自然语言需求转化为可执行的技术任务，能够使用大模型进行需求澄清、代码生成、代码重构、错误定位、测试用例设计和文档生成。该层能力对应真实工程场景中人与智能工具协同完成任务的能力。

第三层是 AI 输出审查能力。学生不仅要会用 AI，更要会审查 AI。审查内容包括代码正确性、算法复杂度、边界条件、安全漏洞、库函数适配、版权风险、数据隐私和模型幻觉。该层能力是本文最核心的教学创新点。

第四层是系统设计与责任能力。学生需要把局部代码能力提升到复杂系统能力，能够解释架构选择、评估工程风险、处理团队协作和理解技术社会后果。大模型越强，人的系统判断与责任意识越不可替代。

4.2 双轨评价：基础能力与人机协同能力并重

在评价制度上，如表 1，本文主张采用“双轨制”。第一轨是无 AI 基础能力评价，主要通过闭卷概念题、限时机试、手写关键算法、现场调试等方式确认学生是否具备基本理解。第二轨是 AI 协同能力评价，要求学生提交 AI 使用说明、提示词日志、版本差异、测试用例、审查报告和口头答辩材料。

双轨评价的目的不是简单增加学生负担，而是把原本不可见的 AI 使用过程转化为可观察、可解释、可评价的学习证据。学生可以使用 AI，但必须说明为什么这样问、为什么采纳或拒绝某个输出、如何验证代码、如何处理边界条件。

5 调研与课堂准实验设计

5.1 研究对象与总体方法

本文采用“问卷调查、课堂准实验、学习日志分析与半结构化访谈”相结合的混合研究方法。大模型对计算机专业教学的影响具有多维性：它既影响学生的学习资源选择、作业完成方式和知识获取路径，也影响教师的教学设计、作业评价和课堂管理方式。因此，单一研究方法难以全面揭示其复杂影响。问卷调查可

表 1 传统计算机教学与大模型协同教学对比

维度	传统计算机教学	大模型时代教学
教学目标	掌握语法、算法和编码规范	基础能力、AI协同、输出审查、系统责任并重
教师角色	知识讲授者、作业批改者	学习任务设计者、AI使用规范制定者、过程证据评价者
学生角色	独立完成代码任务	与AI协同解决问题，并证明自己理解
作业形态	编程题、实验报告、课程设计	提示词日志、AI审查报告、测试报告、代码答辩
评价重点	程序能否运行、结果是否正确	问题分解、推理过程、测试设计、AI输出验证
教学风险	反馈滞后、训练枯燥	依赖AI、浅层复制、学术诚信、能力空心化
考核方式	闭卷考试、机试、实验报告	无AI基础考核、AI协同项目、过程性证据、口头答辩
能力观	会不会写出代码	能否解释、验证、改进和负责

以从较大样本层面把握学生使用大模型的普遍状况与态度倾向；课堂准实验可以比较传统教学与大模型协同教学在学习成效上的差异；学习日志分析能够还原学生使用大模型的真实过程；访谈则有助于进一步解释量化数据背后的原因、动机和风险感知。四种方法相互补充，可以形成较为完整的证据链。

研究对象可覆盖计算机科学与技术专业大一至大四学生，并根据不同年级的课程特点进行分层设计。大一学生主要处于程序设计基础学习阶段，其研究重点可放在大模型是否影响学生语法掌握、基本编程习惯和独立编码能力；大二学生通常学习《数据结构》《面向对象程序设计》等核心课程，适合考察大模型对抽象数据类型、算法理解、调试能力和代码组织能力的影响；大三学生进入《数据库系统》《操作系统》《软件工程》等课程后，可重点观察大模型在复杂系统设计、团队协作、文档生成和工程实践中的作用；大四学生则可结合课程设计、毕业设计和科研训练，

考察大模型对综合创新能力、技术选型、论文写作和学术诚信的影响。通过覆盖不同学习阶段，研究能够避免只从某一门课程或某一年级得出片面结论。

在样本规模方面，问卷调查建议覆盖 300—600 名学生，以保证能够对不同年级、不同能力水平和不同 AI 使用习惯的学生进行比较分析。问卷对象可按年级、课程类型和 AI 使用频率进行分层抽样，既包括高频使用大模型的学生，也包括较少使用或基本不使用大模型的学生。课堂准实验可选择《数据结构》或《面向对象程序设计》中的 2—4 个自然班，样本量控制在 120—200 人左右。考虑到教学现场难以完全随机分组，本文建议采用自然班准实验设计，并在实验开始前通过前测成绩、既有课程成绩、编程基础问卷等方式检验实验组与对照组的初始差异。若两组初始水平存在差异，可在后续数据分析中采用协方差分析等方法进行控制。

访谈对象可包括 20—30 名学生和 8—12 名教师。学生访谈对象应尽量覆盖不同年级、不同成绩水平和不同 AI 使用类型，例如高频依赖型、适度协同型、谨慎使用型和基本不用型学生。教师访谈对象则应覆盖程序设计类课程、专业核心课程、软件工程类课程和毕业设计指导教师，以便了解大模型对不同教学环节造成的影响。访谈重点不宜仅限于“是否支持学生使用 AI”，而应深入到教师如何修改作业、如何判断学生真实能力、如何处理学术诚信问题，以及学生如何理解 AI 生成内容、如何判断 AI 是否出错、是否因 AI 而改变学习策略等问题。

学习日志采集周期建议为 8—12 周，与课堂准实验周期保持一致。实验组学生应定期提交 AI 使用日志，包括使用场景、提示词记录、AI 原始输出、学生修改过程、最终采纳结果、测试验证方式和个人反思。学习日志的价值在于，它能够把学生使用大模型的过程从不可见状态转化为可分析材料。仅仅知道学生“用了 AI”并不能说明问题，关键在于学生如何使用 AI：是直接要求模型完成作业，还是请求模型解释概念；是被动复制代码，还是主动比较多个方案；是接受模型输出，还是设计测试用例进行验证。通过对这些过程性材料进行编码，研究可以进一步区分不同类型的 AI 使用行为及其学习效果差异。

本文的总体研究问题可进一步细化为四个层次。第一，大模型改变了学生哪些学习行为。该问题主要关注学生在课前预习、课堂理解、课后作业、代码调试、实验报告撰写和考试复习等环节中的 AI 使用情况，分析大模型是否改变了学生获取知识、解决问题和完成任务的路径。第二，AI 协同教学与传统教学相比，在学习成效上有何差异。该问题不仅比较最终成绩，还应比较概念理解、代码正确率、调试能力、测试设计能力、问题分解能力和反思能力等多个指标。第三，大模型是否削弱学生基础能力，或促进其高阶计算思维发展。该问题是本文的核心张力所在，因为大模型一方面可能导致学生跳过必要的思考过程，形成依赖；另一方面也可能通过即时反馈、方案比较和代码解释促进学生进行更高层次的问题解决。第四，课程目标、作业形态和评价方式应如何重构。该问题关注研究结果如何转化为教学改革方案，包括哪些任务必须无 AI 完成，哪些任务可以 AI 协同完成，哪些任务必须提交 AI 使用证据，以及如何设计能够评价学生真实能力的新型作业和考核方式。

从变量设计看，本文可将“大模型使用方式”作为重要自变量，将学生学习成效作为因变量，并引入学生原有编程基础、学习动机、课程难度和教师教学方式等控制变量。大模型使用方式不宜简单按“使用”与“不使用”二分，而应进一步区分使用频率、使用场景、提示词质量、AI 输出采纳比例、人工修改程度和验证行为等维度。学习成效也不宜只用期末成绩衡量，而应综合考察基础知识掌握、编程实现能力、调试解释能力、测试设计能力、AI 输出审查能力和系统设计能力。这样的变量设计能够更细致地揭示：究竟是哪一种 AI 使用方式可能促进学习，又是哪一种 AI 使用方式可能导致能力空心化。

在数据分析方面，问卷数据可采用描述性统计、相关分析、差异检验和回归分析，考察不同年级、不同成绩水平和不同 AI 使用频率学生之间的差异。课堂准实验数据可采用前后测比较、组间差异分析和协方差分析，判断 AI 协同教学是否在控制初始水平后仍然产生显著影响。学习日志和提示词记录可采用内容分析、编码分析和序列分析，识别学生与大模型互动的典型模式。访谈资料则可采用主题分析法，提炼学生和教师对大模型教学价值、风险和评价改革的核心看法。通过量化数据与质性材料的相互印证，研究可以避免单纯依赖学生自我报告或单次成绩结果所带来的偏差。

本文采用混合研究方法的目的，不是简单并列多种数据来源，而是通过三角验证提高结论的可靠性。例如，如果问卷显示高频使用 AI 的学生自认为学习效率提高，课堂实验也显示其项目完成速度更快，但后测中的无 AI 编程题表现并未提升，学习日志又显示其大量复制 AI 代码，那么就可以进一步说明“效率提升”与“能力提升”之间存在差异。相反，如果某些学生虽然使用 AI 频率较高，但日志显示其经常进行方案比较、错误追问和测试验证，且后测中调试能力和代码审查能力明显提升，则说明规范化、反思型的大模型使用可能促进高阶能力发展。通过这种多源数据交叉分析，本文可以更准确地回答大模型究竟是在削弱学习，还是在重塑学习。

5.2 问卷调查设计

问卷调查旨在从较大样本层面了解计算机科学与技术专业学生使用大模型的总体状况、典型场景、行为深度、学习效果感知和风险意识，为后续课堂准实验与访谈分析提供基础数据。问卷可设置六个核心维度：AI 使用频率、使用场景、使用深度、学习效果感知、风险感知和 AI 素养。六个维度分别对应“学生是否使用 AI”“在什么任务中使用 AI”“以何种方式使用 AI”“学生认为 AI 带来了哪些帮助”“学生是否意识到潜在风险”以及“学生是否具备规范、批判和负责任使用 AI 的能力”。通过这些维度，可以避免将学生简单区分为“使用 AI”和“不使用 AI”两类，而是进一步刻画其使用方式和学习质量。

第一，AI 使用频率用于测量学生接触和依赖大模型的程度。问卷可设置“从不使用、偶尔使用、每周 1—2 次、几乎每天使用、每天多次使用”等选项，也可进一步询问学生最常使用的大模型工具类型，如 ChatGPT、DeepSeek、通义千问、文心一言、GitHub Copilot、Claude、Gemini 等。除总体频率外，还应区分课程学习、编程作业、课程设计、考试复习、论文写作和日常技术查询等不同场景下的使用频率。这样可以判断学生是否只是在个别任务中偶然使用 AI，还是已经把 AI 作为稳定的学习基础设施。

第二，AI 使用场景用于考察大模型在计算机专业学习中的具体嵌入方式。问卷可将使用场景细分为解释概念、生成代码、调试代码、优化代码、生成测试用例、撰写实验报告、阅读英文技术文档、阅读论文、完成课程设计、准备考试、生成学习计划、解释报错信息等。对于计算机专业而言，还可以进一步加入“理解算法复杂度”“比较不同数据结构方案”“生成单元测试”“分析开源项目代码”“辅助配置开发环境”等更具专业特征的场景。该维度有助于识别大模型主要被学生用于低阶任务，还是已经进入复杂问题解决和工程实践过程。

第三，AI 使用深度用于区分不同质量的人机交互行为。仅统计使用频率并不能说明学生是否真正发生学习，因为高频使用既可能意味着积极探索，也可能意味着过度依赖。使用深度可以分为若干层次：最低层次是直接复制答案，即学生将作业要求输入大模型并直接提交生成结果；较低层次是查询解释，即要求 AI 解释概念、语法或报错信息；中等层次是调试协助，即让 AI 帮助定位错误并提出修改建议；较高层次是方案比较，即要求 AI 给出多个实现思路并分析优缺点；最高层次是审查反思，即学生主动要求 AI 检查边界条件、复杂度、安全风险，并在此基础上进行人工验证。通过这种分层设计，问卷能够揭示学生是在“替代性使用 AI”，还是在“协同性使用 AI”。

第四，学习效果感知用于了解学生主观认为大模型对学习产生的影响。该维度可采用五点李克特量表，例如“非常不同意—不同意—一般—同意—非常同意”。题项可包括：“使用大模型提高了我完成编程作业的效率”“使用大模型帮助我理解抽象概念”“使用大模型提高了我定位程序错误的能力”“使用大模型使我更愿意尝试复杂项目”“使用大模型降低了我学习程序设计的焦虑感”“使用大模型帮助我形成了更清晰的代码结构”。同时，也应设置反向题项，如“使用大模型后，我减少了独立思考的时间”“使用大模型后，我更容易跳过不懂的知识点”“即使代码能够运行，我也不一定理解其实现原理”。正反题项结合有助于提高问卷测量的有效性。

第五，风险感知用于考察学生是否认识到大模型在学习中的潜在负面影响。相关题项可包括：“我担心长期使用大模型会削弱独立编程能力”“我担心 AI 生成代码中存在隐藏错误”“我担心 AI 生成内容可能造成学术不诚信问题”“我担心 AI 生成代码可能存在安全漏洞或版权风险”“我认为 AI 有时会给出看似合理但实际错误的解释”“我认为在课程作业中应明确标注 AI 使用情况”。这一维度对于本文尤为重要，因为计算机专业学生未来可能从事软件开发、系统设计和数据处理工作，若缺乏对模型幻觉、安全漏洞、数据隐私和责任边界的认识，就可能将课堂中的浅层依赖延续到真实工程实践中。

第六，AI 素养用于测量学生是否具备规范、有效和批判性使用大模型的能力。该维度可包括提示词设计能力、输出验证能力、伦理规范意识和反思能力四个方面。提示词设计能力可通过学生是否会明确输入输出要求、约束条件、算法复杂度、编程语言和测试样例来衡量；输出验证能力可通过学生是否会运行代码、设计边界测试、检查复杂度、查阅官方文档或进行人工推理来衡量；伦理规范意识可通过学生是否愿意标注 AI 使用、是否理解学术诚信要求、是否避免上传敏感数据来衡量；反思能力则可通过学生是否会比较 AI 建议与个人思路、总结 AI 错误类型、记录修改依据来衡量。该维度能够直接对应本文提出的“AI 协同能力”和“AI 输出审查能力”。

在问卷结构上，可分为基本信息、AI 使用行为、学习效果感知、风险感知、AI 素养和开放性问题六个部分。基本信息包括年级、已修课程、编程经验、常用编程语言、最近一次相关课程成绩区间、是否参与过课程设计或科研项目等。AI 使用行为部分主要采用单选题和多选题，了解使用频率、工具类型和使用场景。学习效果、风险感知和 AI 素养部分可采用五点李克特量表，以便进行统计分析。开放性问题可设置为：“请描述一次你使用 AI 解决编程问题的经历”“AI 在哪类任务中最容易帮助你”“AI 在哪类任务中最容易误导你”“你认为教师应如何规范学生使用 AI 完成课程作

业”。开放性问题虽然不便于大规模量化，但能够为访谈提纲和课堂实验设计提供更具体的线索。

在问卷设计中，还应注意区分“自我感知能力”和“实际能力”。学生可能认为自己能够有效使用 AI，但未必真正具备判断 AI 输出正确性的能力。因此，除态度类题项外，可适当加入情境判断题。例如，给出一段由 AI 生成的存在边界错误的代码，让学生判断其是否可靠；给出两个提示词，让学生判断哪一个更适合生成可验证的程序；给出一个 AI 解释算法复杂度的回答，让学生判断其中是否存在错误。通过这类小型情境题，可以初步测量学生的 AI 输出审查意识，而不只是依赖学生自我报告。

基于上述问卷维度，本文可提出四个研究假设。H1：高频使用 AI 的学生不一定在无 AI 基础测试中取得更高成绩，但其项目完成效率和任务提交完整度可能更高。该假设旨在区分“完成任务效率”与“基础知识掌握”之间的差异，防止把 AI 带来的速度提升误认为学习能力提升。H2：高质量提示词策略与学生的概念理解、调试能力和测试设计能力呈正相关。其理论依据在于，高质量提示词通常包含明确的问题描述、约束条件、输入输出样例和验证要求，能够反映学生较高的问题建模能力。H3：AI 协同教学对中等水平学生的促进作用可能最为明显，而对基础薄弱学生可能产生更高依赖风险。中等水平学生通常具备一定基础，能够理解并修正 AI 输出；基础薄弱学生则可能因缺少判断能力而直接复制模型结果。H4：在课程任务中加入 AI 输出审查环节后，学生的代码理解深度、边界条件意识和错误识别能力将显著提升。该假设直接服务于本文关于“会审查 AI 比会使用 AI 更重要”的核心观点。

在数据分析方面，首先可对问卷数据进行描述性统计，呈现不同年级学生的 AI 使用频率、主要使用场景和风险感知水平。其次，可通过交叉分析比较不同年级、不同成绩水平、不同使用频率学生在 AI 素养上的差异。例如，可以比较大一学生与大三学生在“AI 输出审查能力”上的差异，也可以比较高频使用者与低频使用者在“直接复制答案”倾向上的差异。再次，可采用相关分析或回归分析检验提示词质量、AI 使用深度与学习效果感知之间的关系。若问卷与课堂成绩、前后测成绩或实验任务表现相结合，还可以进一步分析 AI 使用行为对实际学习结果的预测作用。

需要指出的是，问卷调查的目的并不是简单证明学生使用 AI 越多越好，或使用 AI 越多越差，而是识别不同使用方式背后的学习机制。直接复制型使用可能削弱独立思考，解释查询型使用可能帮助概念理解，调试协作型使用可能提高问题定位效率，审查反思型使用则可能促进高阶计算思维。只有将使用频率、使用场景、使用深度和 AI 素养结合起来分析，才能较为准确地判断大模型对计算机专业学习的真实影响。这样，问卷结果也可以为后续课堂准实验中的任务设计、分组控制和评价量规提供依据。

5.3 课堂准实验设计

实验课程建议选择《数据结构》或《面向对象程序设计》作为准实验载体。这两类课程既包含明确的基础知识目标，又具有较强的编程实践属性，能够较好地观察大模型对概念理解、代码实现、调试能力、问题分解能力和工程表达能力的综合影响。其中，《数据结构》课程更适合考察学生对抽象数据类型、算法复杂度和边界条件的理解；《面向对象程序设计》课程则更适合考察学生在类设计、接口抽象、代码复用和软件结构组织方面的人机协同能力。为保证实验结果具有可比性，可选择同一年级、相近学业基础的 2—4 个自然班作为研究对象，设置对照组和实验组，并在实验开始前通过前测成绩、既有课程成绩或编程基础问卷对两组学生的初始水平进行检验。

对照组采用传统教学模式，主要依靠教材、课堂讲授、课后练习、搜索引擎和教师答疑完成学习任务。教师按照原有课程安排讲授核心概念、布置编程实验并进行结果性评价，学生提交源代码、实验报告和运行结果。实验组则采用大模型协同教学模式，允许学生在规定范围内使用 ChatGPT、DeepSeek、通义千问、GitHub Copilot 等大模型工具辅助学习，但必须遵守明确的使用规范。具体而言，实验组学生不仅需要提交最终代码，还必须同步提交提示词记录、AI 原始输出、人工修改说明、测试用例、错误修复过程和反思日志。通过这一方式，可以把原本隐性的 AI 使用过程转化为可观察、可追踪和可评价的学习证据，避免仅以最终代码判断学生能力。

实验周期建议设置为 8—12 周，并采用“前测—阶段任务—过程跟踪—后测—访谈”的完整流程。前测主要用于了解学生在基本语法、数据结构概念、算法复杂度分析和基础调试方面的初始水平；阶段任务用于持续观察学生在不同任务类型中的表现变化；后测则用于比较两组学生在无 AI 条件下的基础能力和在 AI 协同条件下的高阶能力差异。为减少教师因素对实验结果的影响，两组应尽量使用相同的教学大纲、核心知识点、作业难度和评分标准。不同之处主要体现在实验组允许并规范使用大模型，而对照组仍采用传统学习资源和教师答疑方式。

以《数据结构》课程为例，实验任务可分为四类，并按照由低阶到高阶的顺序递进展开。

第一类是基础实现任务，主要考查学生对基本数据结构和算法思想的掌握情况。任务内容可包括单链表的插入与删除、栈与队列的应用、二叉树遍历、二叉搜索树操作、图的深度优先搜索和广度优先搜索等。

对照组要求学生独立完成代码实现；实验组可以使用大模型辅助理解题意、生成初始代码或解释关键语句，但必须说明 AI 输出中哪些内容被采纳、哪些内容被修改，以及修改的依据。该类任务的评价重点不是简单比较代码是否运行成功，而是考查学生是否真正理解数据结构的基本操作、时间复杂度和典型边界情况。

第二类是调试任务，主要考查学生的问题定位和错误解释能力。教师可提供若干段包含隐藏错误的程序，例如链表空指针错误、递归终止条件错误、树遍历顺序错误、图遍历中访问标记设置错误、优先队列使用不当等。学生需要定位错误、修复代码并解释错误产生的原因。实验组允许使用大模型辅助分析错误，但必须记录与模型的交互过程，并说明最终判断是否完全来自 AI 建议，还是经过了人工验证。该类任务有助于区分学生是机械接受 AI 答案，还是能够结合运行结果、调试信息和理论知识进行独立判断。

第三类是 AI 审查任务，主要考查学生对大模型生成代码的批判性评价能力。教师可事先使用大模型生成若干段“表面正确但存在隐蔽问题”的代码，如只适用于一般输入而忽略边界条件的排序算法、复杂度不符合要求的图算法、缺少异常处理的链表操作、存在死循环风险的递归程序，或在大规模数据下性能明显下降的实现。学生需要从正确性、复杂度、鲁棒性、安全性、可读性和边界条件等角度对代码进行审查，并提交审查报告。该类任务是实验组教学设计的关键环节，因为它能够直接检验学生是否具备“会使用 AI”之外更重要的“会审查 AI”的能力。

第四类是综合项目任务，主要考查学生的问题建模、系统设计和综合应用能力。项目可选择“校园路径规划系统”“课程作业自动评测系统”“图书借阅管理系统中的数据索引模块”或“基于树结构的表达式求值系统”等。任务要求学生完成需求分析、数据结构选择、算法设计、模块划分、代码实现、测试设计和结果说明。实验组可以在需求澄清、方案比较、代码重构、测试用例生成和文档撰写中使用大模型，但必须提交完整的 AI 协同过程记录。教师在评价时不仅关注系统是否完成，还应重点考查学生是否能够解释为什么选择某种数据结构，如何验证算法正确性，如何处理异常输入，以及如何判断 AI 建议是否合理。

为增强实验的可操作性，可进一步将每类任务设置为“无 AI 完成”和“AI 协同完成”两个层次。基础实现任务和核心概念测试应保留一定比例的无 AI 环节，用于确保学生具备必要的基础能力；调试任务、AI 审查任务和综合项目任务则可以更多引入 AI 协同，以考查学生在真实人机协作环境中的问题解决能力。这样的设计既避免学生对大模型形成过度依赖，也避免把 AI 简单排除在教学之外，从而更符合当前计算机专业学习和软件开发实践的真实情境。

在数据采集方面，课堂准实验不应只记录最终成绩，还应收集多源过程性数据。除前测和后测成绩外，还可采集代码提交记录、自动评测结果、测试用例数量与覆盖情况、提示词迭代次数、AI 输出采纳比例、人工修改比例、调试耗时、审查报告质量和学生反思日志等。对于实验组，还可以对提示词进行编码，区分直接求答案型、概念解释型、调试定位型、方案比较型和审查反思型，并进一步分析不同类型 AI 使用行为与学习成效之间的关系。对于对照组，则可记录搜索引擎使用、课堂提问、教师答疑和代码修改过程，以比较传统学习支持与 AI 学习支持在反馈效率、认知参与和结果质量上的差异。

表 2　课堂准实验评价指标与数据来源

指标类型	具体指标	分析方法
知识掌握	前测/后测成绩、概念题正确率	t检验、协方差分析
编程能力	代码正确率、测试通过率、代码复杂度	自动评测、代码静态分析
调试能力	错误定位时间、错误解释质量	过程日志、评分量规
计算思维	问题分解、抽象建模、算法选择	专家评分、同伴互评
AI 协同能力	提示词质量、迭代次数、AI输出采纳方式	提示词编码、序列分析
审查能力	发现AI代码错误、边界问题和安全问题的数量	错误清单、审查报告评分
学习过程	Git提交记录、AI对话日志、反思日志	日志挖掘、主题分析
学术诚信	AI使用标注、提交内容解释能力	答辩记录、诚信声明

在评价方式上，建议采用统一评分量规，避免仅凭教师主观印象判断实验效果。如表 2，评分量规可包括六个维度：概念理解、代码正确性、复杂度分析、调试解释、测试设计和反思质量。实验组还应增加 AI 协同能力和 AI 输出审查能力两个维度，具体考查提示词是否明确、是否包含约束条件、是否对 AI 输出进行验证、是否能够发现模型生成代码中的错误，以及是否能够说明最终方案与 AI 建议之间的关系。通过这些指标，研究可以更准确地区分“AI 提高了任务完成效率”与“AI 真正促进了学习能力发展”之间的差异。

通过上述准实验设计，本文能够比较传统教学与大模型协同教学在不同能力维度上的差异。预期结果并不应简单理解为实验组一定全面优于对照组。更可能出现的情况是：实验组在任务完成效率、代码重构、测试生成和项目表达方面表现更好，但在无 AI 条件下

的基础概念掌握和独立编码能力方面可能存在差异；而对照组在基础训练方面更稳定，但在复杂项目迭代、方案比较和即时反馈利用方面可能受到限制。正是这种差异，能够为本文提出的“双轨评价”提供实证依据，即计算机专业教学既要坚持无 AI 条件下的基础能力训练，又要面向真实工程环境培养学生规范使用、审查和反思大模型的能力。

5.4 学习日志与提示词分析

学习日志是本文实证设计的关键创新点。实验组学生需提交 AI 对话记录节选、提示词修改过程、AI 生成代码与最终代码差异、采纳或拒绝 AI 输出的理由、测试用例设计记录和反思日志。通过日志可以识别学生到底是在“复制答案”，还是在“借助 AI 进行解释、比较、验证和反思”。

如表 3，提示词可编码为五类：直接求答案型、概念解释型、调试定位型、方案比较型和审查反思型。进一步可测量 AI 依赖度，包括 AI 生成代码占最终代码比例、学生修改比例、学生解释准确率和边界测试覆盖度。该分析能够把 AI 使用从不可见行为转化为可量化证据。

表 3 AI协同学习行为分类模型

类型	行为特征	学习风险	教学干预
复制型	直接让AI完成作业，较少追问	高依赖、低理解	增加口头答辩和代码解释
查询型	让AI解释概念或语法	有助理解但深度有限	要求举例、迁移和反例分析
调试型	用AI定位错误并修改代码	有助实践但可能外包思考	要求说明错误原因和修复依据
协作型	与AI迭代设计方案并比较取舍	学习深度较高	鼓励方案比较和测试覆盖
审查型	主动质疑AI输出并验证边界	最符合高阶能力目标	纳入核心评价量规

5.5 访谈设计与三角验证

学生访谈重点包括：在哪些课程中最常使用大模型；是否直接复制 AI 代码及其原因；AI 是帮助理解还是只是加快完成作业；哪些任务中 AI 最有效，哪些任务中 AI 最容易误导；如果教师要求提交提示词和 AI 使用说明，学习方式是否会改变。

教师访谈重点包括：原有作业是否仍能评价学生真实能力；教师是否调整实验题、课程设计和考试方式；哪些能力必须坚持无 AI 训练；是否愿意把 AI 输出审查纳入课程目标；学校是否有明确的 AI 使用规范。问卷、实验成绩、日志编码和访谈材料可相互印证，形成更稳健的证据链。

5.6 模型应用的预判分析与实施策略

在实际应用中，建议采用“课程试点—跨课扩展—制度固化”的渐进路径。第一阶段可选择《程序设计基础》或《数据结构》的若干自然班开展小规模试点，在保持课程目标、教学进度和考核要求一致的前提下，将无 AI 基础考核与 AI 协同项目同时纳入课程，先检验任务难度、日志提交和评分量规的可操作性。第二阶段可将经试点修订的任务模板扩展至软件工程、数据库系统和毕业设计等课程，形成由基础训练、专业核心课到综合实践的递进安排。第三阶段再把 AI 使用说明、输出审查报告和现场答辩纳入课程规范，使四层能力模型由单门课程尝试转化为专业培养方案中的稳定要求。

从应用效果看，可作出审慎预判：AI 协同轨可能首先改善任务完成效率、错误定位、测试设计和项目表达，但这些变化不必然同步转化为无 AI 条件下的概念掌握和独立编程能力。基础较弱的学生可能因缺少判断依据而更容易直接采纳模型输出，具备一定基础的学生则更可能利用 AI 开展方案比较、调试和审查。因此，实施中不宜只比较总成绩，而应分别观察基础能力、协同效率、输出审查和系统责任四个维度，并重点分析不同基础水平学生的收益与依赖风险。上述判断属于应用前的分析，仍需通过课堂数据加以验证。

为使预判能够被检验，建议建立“前测—任务实施—过程记录—后测与答辩—反馈修订”的闭环。前测用于确认实验班与对照班的初始差异；任务实施阶段统一教学内容和评分量规；过程记录保留提示词、代码版本、测试用例、采纳理由和教师观察；后测同时设置无 AI 基础任务与 AI 协同审查任务，并结合访谈解释结果。教师团队应定期开展评分校准，学校层面则需明确数据隐私、学术诚信、工具可及性和 AI 使用标注要求。若某项安排导致基础能力下降、学生依赖增加或教师评价负担明显失衡，应据此调整 AI 开放范围、任务比例和证据要求，而不是预设模型必然有效。

6 结束语

大模型对计算机科学与技术专业教学的影响具有结构性和不可逆性。它既削弱了传统编程作业作为能力评价工具的有效性，也为个性化反馈、即时调试和复杂项目实践提供了新的可能。问题不在于是否允许学生使用 AI，而在于是否能够把 AI 使用纳入清晰的教学目标、过程记录和评价制度之中。

本文提出的“基础能力—AI 协同—输出审查—系统责任”四层能力模型，试图回应大模型时代计算机专业“还教什么、怎样教、如何评”的核心问题。未来教

学改革应坚持双轨评价：基础训练阶段强调无 AI 条件下的概念、算法和调试能力；高阶实践阶段则面向真实人机协同环境，培养学生的问题建模、提示设计、输出审查、测试验证和工程责任能力。

后续工作可从三个方面继续推进：一是将该能力模型嵌入程序设计基础、数据结构、软件工程和毕业设计等不同阶段课程，形成分层递进的 AI 使用规范与评价量规；二是持续积累提示词日志、代码版本、测试记录和答辩材料，建立可复核的学习过程数据集，以检验不同类型 AI 使用行为与学习成效之间的关系；三是加强教师培训和制度建设，明确哪些任务必须无 AI 完成、哪些任务允许 AI 协同、哪些任务必须提交审查证据。通过持续迭代，计算机专业教学才能从“禁止 AI 或放任 AI”的二元选择，转向“规范 AI、审查 AI、反思 AI、负责地使用 AI”的专业教育新范式。

参考文献

[1] Raihan N, Siddiq M L, Santos J C S, et al. Large Language Models in Computer Science Education: A Systematic Literature Review[C]//Proceedings of the 56th ACM Technical Symposium on Computer Science Education V.1. New York: ACM, 2025: 938-944.

[2] Denny P, Leinonen J, Prather J, et al. Prompt Problems: A New Programming Exercise for the Generative AI Era[C]//Proceedings of the 55th ACM Technical Symposium on Computer Science Education V.1. New York: ACM, 2024: 296-302.

[3] 徐悦, 黄子文, 宋雨轩, 等. 从 AI 大模型看高校计算机教育面临的机遇与挑战[J]. 计算机技术与教育学报, 2024, 12(3).

[4] Raihan N, Goswami D, Puspo S S C, et al. On the Performance of Large Language Models on Introductory Programming Assignments[J]. Journal of Intelligent Information Systems, 2026, 64: 239-263.

[5] Freeman J. Student Generative AI Survey 2025[R/OL]. Oxford: Higher Education Policy Institute, 2025[2026-05-19]. https://www.hepi.ac.uk/reports/student-generative-ai-survey-2025/.

[6] Stephenson R, Armstrong C. Student Generative Artificial Intelligence Survey 2026[R/OL]. Oxford: Higher Education Policy Institute, 2026[2026-05-19]. https://www.hepi.ac.uk/reports/student-generative-ai-survey-2026/.

[7] Lepp M, Kaimre J. Does Generative AI Help in Learning Programming: Students' Perceptions, Reported Use and Relation to Performance[J]. Computers in Human Behavior Reports, 2025, 18: 100642.

[8] Jošt G, Taneski V, Karakatič S. The Impact of Large Language Models on Programming Education and Student Learning Outcomes[J]. Applied Sciences, 2024, 14(10): 4115.

[9] Liu R, Zenke C, Liu C, et al. Teaching CS50 with AI: Leveraging Generative Artificial Intelligence in Computer Science Education[C]//Proceedings of the 55th ACM Technical Symposium on Computer Science Education V.1. New York: ACM, 2024: 750-756.

[10] Liu R, Zhao J, Xu B, et al. Improving AI in CS50: Leveraging Human Feedback for Better Learning[C]//Proceedings of the 56th ACM Technical Symposium on Computer Science Education V.1. New York: ACM, 2025: 715-721.

[11] 李志刚, 杨吉斌, 张睿, 等. 基于 ChatGPT 的程序设计翻转课堂教学方法实践[J]. 计算机技术与教育学报, 2023, 11(2).

[12] Xue Y, Chen H, Bai G R, et al. Does ChatGPT Help with Introductory Programming? An Experiment of Students Using ChatGPT in CS1[C]//Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training. New York: ACM, 2024: 331-341.

[13] Gambo O, Onifade O J, Olanrewaju-Smart W, et al. Impact of ChatGPT on Learning Outcomes and Performance of Students in Computer Programming Courses: A Mixed-Method Approach[J]. Discover Education, 2026, 5: 164.

[14] Mutanga M B, Msane J, Mndaweni T N, et al. Exploring the Impact of LLM Prompting on Students' Learning[J]. Trends in Higher Education, 2025, 4(3): 31.

[15] Ding L, O'Berry R, Tallent H, et al. Learning with Large Language Models: Beyond Prompt Engineering[J]. Education and Information Technologies, 2026. DOI: 10.1007/s10639-026-13924-2.

[16] 王兵书, 李静怡, 雍珊珊, 等. 基于提示工程的程序设计探索与实践[J]. 计算机技术与教育学报, 2024, 12(3): 165-171.

[17] ACM, IEEE Computer Society, AAAI. Computer Science Curricula 2023: Curricular Guidelines for Undergraduate Degree Programs in Computer Science[EB/OL]. (2024-06-05). https://csed.acm.org/.

[18] Eaton E, Epstein S L. Artificial Intelligence in the CS2023 Undergraduate Computer Science Curriculum: Rationale and Challenges[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2024, 38(21): 23078-23083.

[19] 邹雄, 刘宇航, 刘栓, 等. 人工智能技术对高校人才培养的影响[J]. 计算机技术与教育学报, 2024, 12(4): 83-86.

[20] Jing Y, Wang H, Chen X, et al. What Factors Will Affect the Effectiveness of Using ChatGPT to Solve Programming Problems? A Quasi-Experimental Study[J]. Humanities and Social Sciences Communications, 2024, 11: 319.