Advances in Education 教育进展, 2026, 16(11), 1504-1513
Published Online November 2026 in Hans. https://www.hanspub.org/journal/ae
https://doi.org/10.12677/ae.2026.16112529

Hans汉斯

农林类高职院校软件技术专业引入人工智能辅助编程教学的实践探索

李　嘉

上海农林职业技术学院智慧农业工程系，上海

收稿日期：2026年10月11日；录用日期：2026年11月12日；发布日期：2026年11月19日

摘　要

生成式人工智能推动编程活动由“逐行手写”转向“人机协同”。对基础薄弱、重实操的农林类高职软件技术专业学生而言，AI辅助编程工具既是降低认知负荷的“认知脚手架”，也可能诱发认知卸载与工具依赖。本文以上海某农林高职软件技术专业为实践载体，在6门核心课程中立足智慧农业人才需求，对AI辅助编程教学进行本土化改造，形成“教师引导 + AI辅助 + 项目实操”模式。经两个学期教学对比实验并结合问卷访谈，从认知负荷与自我调节学习理论视角剖析其成效与风险：实验班代码完成率与程序正确率较对照班分别提升22.3与21.5个百分点，学习信心显著增强，但也暴露工具依赖、农林实训资源不足、评价成本偏高等问题，研究据此提出有计划淡出、技术辅助评价、微型项目起步等对策。研究认为，农林类高职院校引入AI辅助编程教学的关键不在于创建全新模式，而在于面向智慧农业人才需求的系统性本土化改造，可为同类院校提供参考。

关键词

农林类高职院校，软件技术，AI辅助编程，认知负荷，本土化改造

Practical Exploration of Introducing AI-Assisted Programming Teaching into the Software Technology Major of Agriculture and Forestry Higher Vocational Colleges

Jia Li

Department of Smart Agricultural Engineering, Shanghai Vocational College of Agriculture and Forestry, Shanghai

文章引用: 李嘉. 农林类高职院校软件技术专业引入人工智能辅助编程教学的实践探索[J]. 教育进展, 2026, 16(11): 1504-1513. DOI: 10.12677/ae.2026.16112529

Received: October 11, 2026; accepted: November 12, 2026; published: November 19, 2026

Abstract

Generative artificial intelligence (AI) is transforming programming from line-by-line manual coding toward human-AI collaboration. For students in the software technology major of agriculture and forestry higher vocational colleges, who tend to have weak foundations and a strong orientation toward hands-on practice, AI-assisted programming tools act both as a cognitive scaffold that lowers cognitive load and as a potential trigger of cognitive offloading and tool dependence. Taking the software technology major of an agriculture and forestry higher vocational college in Shanghai as the practice setting, this study conducted localized adaptation of AI-assisted programming teaching in six core courses in line with the talent demands of smart agriculture, and formed a "teacher guidance + AI assistance + project practice" model. Based on a two-semester comparative teaching experiment combined with questionnaires and interviews, the effects and risks were analyzed from the perspectives of cognitive load theory and self-regulated learning: the code completion rate and program correctness rate of the experimental class increased by 22.3 and 21.5 percentage points respectively over the control class, and learning confidence improved significantly; meanwhile, problems such as tool dependence, insufficient agriculture and forestry training resources, and high evaluation costs were also exposed. Accordingly, countermeasures including planned fading of AI support, technology-assisted evaluation, and starting with micro-projects are proposed. The study concludes that the key to introducing AI-assisted programming teaching in agriculture and forestry higher vocational colleges lies not in creating an entirely new model, but in systematic localized adaptation oriented toward the talent demands of smart agriculture, which can serve as a reference for similar colleges.

Keywords

Agriculture and Forestry Higher Vocational Colleges, Software Technology, AI-Assisted Programming, Cognitive Load, Localized Adaptation

Copyright © 2026 by author(s) and Hans Publishers Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

1. 引言

生成式人工智能的快速普及正在重塑软件行业的工作方式。GitHub Copilot、Cursor、CodeLlama 等 AI 辅助编程工具被广泛用于代码编写、调试与优化，AI 协同开发已成为企业开发岗位的普遍要求[1]-[3]。与此同时，数字乡村与智慧农业战略深入推进，农业物联网、农业信息管理系统等岗位既要求从业者具备基础编程能力，也要求其能够借助 AI 工具提升开发效率。农林类高职院校软件技术专业作为面向区域农业数字化的复合型技术技能人才培养载体，其课程教学必须主动适应这一变化。

从学情看，高职软件技术专业学生普遍存在计算机逻辑基础薄弱、编程入门困难、代码调试耗时长等突出问题。传统编程课堂班级规模较大，教师难以兼顾每位学生的个性化排错答疑，学生易产生畏难情绪与挫败感，学习信心不足。从产业端看，软件岗位已不再要求开发者从零手写全部代码，能够借助 AI 工具辅助开发、校验与改写代码已成为岗位的基本要求。职业院校若固守传统教学方式，将导致学生能力结构与未来就业需求相脱节。

梳理已有研究可以发现，国内关于 AI 辅助编程教学的研究近年快速增长，但整体仍处于探索期。现

有研究大致沿两条线索展开：其一是工具应用层面，多集中于 Java、Python 等单一课程的试点，探讨 AI 代码助手在答疑、备课、纠错等环节的辅助效果，但鲜见课前–课中–课后一体化的完整教学设计[3]-[6]；其二是教学效果层面，国际上已有实证研究关注 AI 代码生成工具对编程初学者的支持作用及其潜在风险，指出 AI 既能降低入门门槛，也可能诱导学习者跳过必要的认知加工[7]。然而，上述研究存在三点不足：第一，研究对象多面向普通高校与理工类院校，对农林类高职院校“基础薄弱、重实操、强技能”的学情关注不足；第二，多数研究把 AI 辅助编程当作一项通用技术应用来讨论，缺少把 AI 编程能力与特定行业(如智慧农业)人才需求相对接的情境化设计；第三，已有方案往往止于“引入工具”，对引入之后如何在制度、资源、评价层面进行系统性配套改造论述不够充分。

正是基于这一研究缺口，本文的研究定位并非宣称创建某种全新的普适教学模式，而是聚焦一个更具体的问题：在农林类高职院校这一特定情境下，如何对 AI 辅助编程教学模式进行本土化改造，使其对接智慧农业人才需求。具体而言，本文探究两个核心问题：其一，农林类高职院校软件技术专业如何构建适配自身办学定位的 AI 辅助编程教学实施方案；其二，如何实现 AI 工具赋能与学生编程逻辑思维、自我调节学习能力培养之间的平衡。研究综合采用文献研究、教学对比实验、问卷调查与半结构化访谈、数据统计分析等方法，形成可落地、可推广的教学改革实践，以期为同类农林高职院校提供参考。

2. 农林类高职软件技术专业推行 AI 辅助编程教学的多重逻辑

在讨论具体方案之前，需要先厘清农林类高职院校软件技术专业推行 AI 辅助编程教学的内在逻辑。这种逻辑不仅是“技术进课堂”的简单理由，更包含产业、教学、办学定位三层递进关系，以及农林院校特有的约束条件。

2.1. 产业逻辑：软件开发范式转向人机协同

软件行业的生产方式正在发生结构性变化。开发者从“逐行手写代码”转向“人机协同开发”：由 AI 工具辅助完成代码生成、提示补全、错误诊断与优化建议，开发者则负责需求分析、架构设计、代码审核与业务逻辑把关。这一范式要求职业教育的人才培养目标相应迭代。若课程仍仅训练手写代码，学生毕业后将难以适应真实的开发工作流。将 AI 协同开发能力纳入软件技术专业核心技能培养，是职业教育对接产业岗位能力迭代的必然要求[5]。

2.2. 教学逻辑：AI 作为“认知脚手架”的双重效应与理论阐释

AI 辅助编程工具进入课堂，最直接的教学价值在于降低编程入门的认知门槛。这一判断可以借助认知负荷理论(Cognitive Load Theory, CLT)加以阐释。该理论认为，人类的认知加工受限于有限的工作记忆容量；当学习材料或任务本身的内在认知负荷较高，而教学呈现方式又叠加了额外的外在认知负荷时，工作记忆过载，学习受阻[8]。对高职编程初学者而言，编程学习的内在认知负荷本已较高——既要理解语法规则，又要掌握算法逻辑与调试方法；传统课堂中，编译报错信息的专业晦涩、求助教师的等待时间，又构成了显著的外在认知负荷。AI 辅助编程工具通过即时提供语法纠错、报错解释与思路提示，实质上是削减了外在认知负荷，使学生能把有限的工作记忆资源集中投入到对编程逻辑的理解上，从而缓解入门阶段的过载与挫败感。

同时，维果茨基的“最近发展区”理论为 AI 工具的教学定位提供了另一层阐释——AI 辅助编程工具可充当一种“认知脚手架”，在学生遭遇语法错误、逻辑缺陷时提供即时的代码提示、错误诊断与思路解释，帮助其越过凭借自身能力暂时无法跨越的障碍[9]。

然而，仅从“降负”“支撑”的单向视角理解 AI 的认知脚手架作用是不够的。从自我调节学习理论(Self-Regulated Learning, SRL)来看，学习是一个包含“目标设定–策略选择–过程监控–结果反思”的

循环过程，学习者能否主动完成这一循环，决定了学习是走向深度掌握还是流于表面[10]。AI 的引入对这一循环构成了双重效应，学界对此已有清醒的讨论[11] [12]：

其一，正向的“支架循环”：当 AI 被用于验证思路、对比方案、解释报错时，它降低了外在认知负荷，支撑了学生的认知加工，有助于学生建立自己的编程图式，走向真正的领域掌握。

其二，负向的“卸载循环”：当学生不加检验地直接接收 AI 生成的完整代码时，必要的认知挣扎被绕过，任务虽表面完成，但关联加工被压缩，学生形成的是对工具的依赖而非对知识的建构，产生“元认知卸载(metacognitive offloading)”甚至“理解错觉”——学生高估自己对代码的理解程度，却在脱离工具时无法独立解决问题[11] [12]。

因此，AI 作为认知脚手架的本质是“支撑”而非“替代”，其双重效应的走向，取决于教学活动中如何设计任务与规则，引导学生更多进入“支架循环”、减少“卸载循环”。这一理论认识，构成本文教学设计与评价设计的逻辑起点，也是后续问题归因(见 4.3)与优化路径(见 5.1、5.4)的理论依据。

2.3. 办学定位逻辑：农林类高职的差异化人才诉求

农林类高职院校有别于普通理工类院校，其软件技术专业并非单纯培养互联网开发人员，而是面向农业物联网、农业信息系统、智慧城市移动端等涉农行业岗位输送复合型技能人才。这意味着编程教学不能仅训练通用互联网项目，还须适度结合农业数据采集、农业信息展示、农业设备控制等业务场景。将 AI 编程能力与农业数字化业务相结合，既是农林类高职院校办学的差异化优势所在，也是其区别于理工类院校课程改革的特色所在。若简单照搬理工类院校的 AI 编程教学方案，将弱化自身的人才培养特色。

2.4. 农林高职实施 AI 编程教学的特有约束条件

农林类高职院校在推行 AI 辅助编程教学时，面临几重固有约束：一是资源约束，机房网络环境对部分 AI 工具的访问存在限制，付费专业工具的经费预算有限；二是课程约束，总课时有限，既要完成课标规定的通用编程教学内容，又要嵌入智慧农业项目，课时紧张；三是学情约束，部分学生信息素养参差不齐，对 AI 生成内容的辨别能力较弱，容易产生工具依赖；四是资源短板，面向智慧农业的编程实训案例、教学素材储备不足，制约农林特色场景的落地。上述约束既是后续教学实践中各类问题的前置背景，也是本方案设计必须回应的现实条件。

3. 研究设计与实践方案：基于“教师引导 + AI 辅助 + 项目实操”模式的多课程本土化改造

在多重逻辑与约束条件的分析基础上，研究设计并实施了一套面向智慧农业人才需求的本土化教学改革方案。

3.1. 实践基础与课程范围

实践选取软件技术专业 6 门核心课程开展改革，各课程均设置实验班与对照班，选取基础水平相近的平行班以保证实验可比性，开展为期两个学期的对比教学实验。

从各课程教学标准看，6 门课程学时均在 54~64 学时之间，讲授与实训约各占一半，普遍采用“项目引领、任务驱动”的组织模式，为 AI 辅助编程的融入提供了良好的结构基础。例如 Python 程序设计课程(54 学时)设置了“销售数据报告生成系统”综合项目，涵盖 pandas 数据读取、matplotlib 可视化；移动互联开发课程(64 学时)设置“智慧农业移动应用开发”综合实训，需连接智能农业网关获取采集器数据并操作智能农业设备；Java 程序设计课程/面向对象程序设计(64 学时)的实训任务包含绘制作物价格趋势折线图；HTML + CSS 课程(64 学时)以“百度”“51 宿舍”“小米商城”等整页制作为实训载体，对接

1 + X 证书；Linux 操作系统课程(64 学时)涵盖 Shell 脚本编写与 LAMP 服务器部署。

3.2. AI 辅助编程教学工具的遴选机制

研究建立面向高职课堂的工具遴选指标体系，从教学适配性、部署成本(免费/付费)、中文友好度、代码可靠性、网络环境适配、版权与教学伦理风险等维度对候选工具进行测评。在此基础上实行分层使用策略：日常基础教学优先选用可免费部署的开源代码模型与网页版 AI 代码助手，降低课堂经费压力；综合实训项目阶段按需采购 GitHub Copilot、Cursor 等付费专业工具，追求更稳定、更适配教学的服务能力；不同课程按技术栈与教学目标差异化配置工具。工具清单定期测评更新，并同步完善机房网络环境保障。

3.3. “教师引导 + AI 辅助 + 项目实操”模式的内在运行机理

本模式并非教师讲授、学生借助 AI 代写代码的简单组合，而是明确了教师、AI 与学生三方角色的边界与协同关系。教师作为知识建构者、边界管控者与评价主导者，负责明确课堂任务与知识目标、划定 AI 使用边界、讲解代码原理并开展结果审核，防止学生直接复制 AI 生成代码。AI 作为认知脚手架承担辅助职能，提供代码提示、语法纠错、代码解释与优化建议；学生遇到报错时可先向 AI 寻求思路提示，但须理解所写代码的逻辑。项目实操作为落脚点，将课程任务拆解为阶梯式实训任务，在通用编程项目之外适度融入智慧农业轻量化行业项目，如 Python 读取模拟农业传感器数据、HTML 制作农产品展示页面、Java 绘制作物价格趋势图等，实现 AI 编程能力与农业数字化业务的结合。

3.4. 全流程教学设计：课前－课中－课后闭环

构建课前－课中－课后闭环。课前，教师完成 AI 工具测试、任务单设计与实训案例开发，并预设课堂提问示例，确保课堂工具可用；课中，按照“任务发布－自主思考－AI 获取思路提示－学生改写调试－教师点评复盘”的流程组织教学，AI 仅提供思路提示而非完整代码；课后，明确 AI 使用边界，留存调试记录、AI 对话记录、代码注释与修改过程等过程材料，作为评价与防依赖的依据。

在上述闭环中，为落实 2.2 节“引导学生进入支架循环、减少卸载循环”的理论要求，教学设计在三个环节刻意嵌入了防依赖机制：其一，任务发布后要求学生先“自主思考”并简要陈述思路，再接触 AI，从流程上阻断“直接索要代码”的冲动；其二，规定 AI 仅提供思路提示而非完整代码，并鼓励学生向 AI 进行“对比式提问”(如要求 AI 给出多套方案并分析利弊)，把提示词从“直接拿代码”转化为“开展对比思考”；其三，在关键子任务(如编写测试用例、逻辑推演)上设置“关闭 AI 独立完成”的检查点，帮助学生校准对自身真实能力的认知，避免形成“理解错觉”[12]。

3.5. 配套评价体系设计

针对传统“只看程序能否运行”评价的弊端，构建面向 AI 辅助学习的过程性评价体系，从知识掌握度、代码改写与注释能力、与 AI 交互的提问设计能力、调试排错能力、业务逻辑理解、项目完成质量等维度综合评判学生能力，重点考核学生对 AI 生成代码的理解、改写与修改，而非单纯以运行结果论定。评价结合各课程既有的多元考核结构(如移动互联开发课程的“形成性 10% + 育人目标 30% + 项目化 60%”、Java 课程的“形成性 30%+ 育人目标 30%+ 终结性 40%”)，将过程性指标嵌入原有考核框架，避免额外增加过重负担。同时，为支撑过程性评价的落地，研究在部分课程引入代码版本控制工具(如 Git)作为学习过程的留痕载体，要求学生以规范的分步提交记录代码演进过程，为后续借助日志分析辅助评价奠定基础(详见 5.4 节)。

4. 结果与讨论：实践成效与问题归因分析

经过两个学期的教学对比实验，结合成绩数据、问卷与访谈反馈，本实践取得了一定成效，也暴露

出若干问题。本章既报告结果，更着重对问题进行深层归因分析。

4.1. 实验数据结果概览

从客观量化数据看，实验班在代码完成率、程序正确率上较对照班有明显提升，代码调试耗时明显缩短，部分课程任务的一次性通过率提高，其中代码完成率与程序正确率的提升幅度均超过课题设定的20%技术指标(实验班代码完成率 83.5%、程序正确率 76.3%，较对照班分别提升 22.3 与 21.5 个百分点，见图 1)。从主观质性数据看，问卷显示实验班学生在编程学习信心、畏难情绪缓解方面有明显改善，认为 AI 工具帮助自己更快理解了报错原因、减少了求助等待时间(实验班问卷三项均分 4.1~4.3，对照班 2.8~3.0，见图 2)；师生半结构化访谈反馈也印证了上述趋势，多数教师反映 AI 辅助缓解了个性化辅导压力，使其能更专注于重难点讲解与项目思路指导。实践同时形成了覆盖 6 门课程的可操作教学流程、课堂任务单与实训案例，其中包含少量智慧农业相关实训素材，为教学改革成果的固化推广奠定了基础。

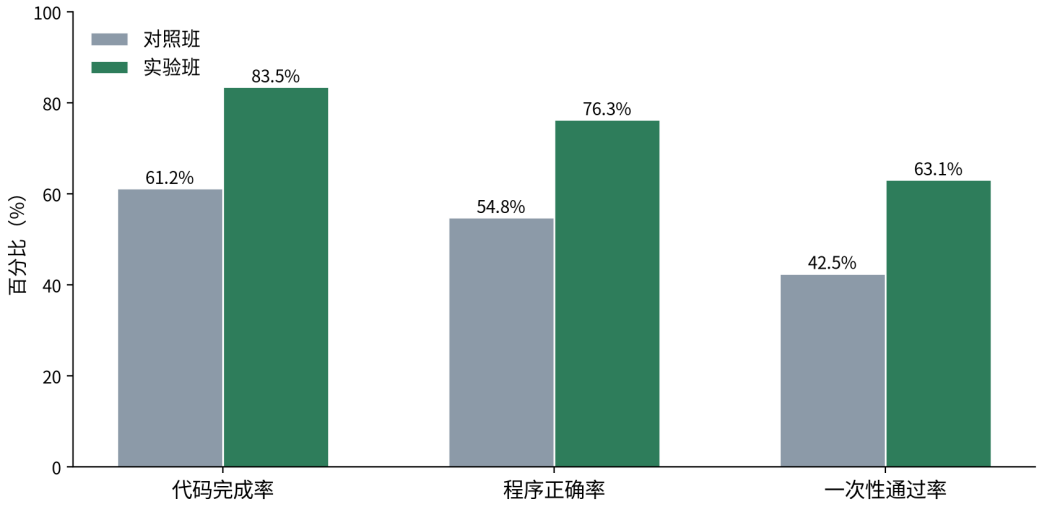


Figure 1. Comparison of core quantitative indicators between the experimental class and the control class
图 1. 实验班与对照班核心量化指标对比

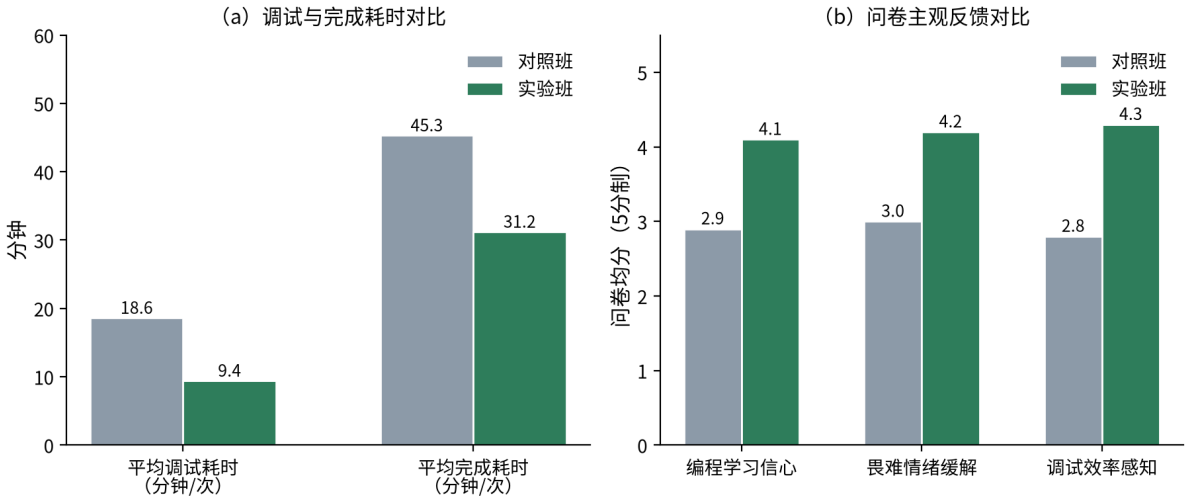


Figure 2. Comparison of debugging time and questionnaire feedback between the experimental class and the control class
图 2. 实验班与对照班调试耗时与问卷反馈对比

4.2. 实践中暴露的核心问题

在取得成效的同时，实践亦暴露了四类突出问题。一是学生层面，部分学生过度依赖 AI 直接生成完整代码，仅以程序能够运行作为完成标准，忽视了代码逻辑的理解，编程逻辑推演能力与自我调节学习能力被弱化。二是教学资源层面，智慧农业场景的实训案例数量偏少，多数仍为通用软件案例，农林办学特色难以充分体现。三是工具与环境层面，付费工具存在经费约束，部分机房网络环境对个别 AI 工具的访问不稳定，影响课堂连续性。四是评价执行层面，AI 生成作业与真实能力难以甄别，过程评价落地执行成本较高，增加教师工作量。

4.3. 问题深层归因分析

上述问题并非孤立现象，其背后存在多重叠加的深层原因，可从理论视角进一步厘清。

学生过度依赖 AI，根源不限于学生个体的自觉性问题，而是学情、制度与评价三重因素叠加，本质上对应 2.2 节所述的“卸载循环”被不当强化。其一，高职学生编程畏难心理驱动其倾向于直接获取完整代码而非自主思考；其二，课堂初期缺少明确的 AI 使用制度边界，未提前规定 AI 的角色定位与使用禁区；其三，原有考核方式仅以运行结果作为评判标准，客观上变相激励了“程序能够运行即可”的取向，削弱了深度思考的动机。从自我调节学习视角看，“只问结果、不问过程”的评价导向，恰恰剥夺了学生在“监控–反思”环节的元认知投入，使其自我调节学习能力难以在真实的任务挣扎中得到锻炼。可见，依赖问题根植于教学制度与评价导向，需从制度与评价层面加以化解。

农林特色实训资源不足，主要源于软件课程教师缺乏农业行业业务背景，且缺少与农业专业教师的跨专业协同教研机制，导致智慧农业案例开发能力有限、供给不足。更深层地看，这一问题是“技术能力”与“行业场景知识”分属两个教师群体所致，若无制度化的协同载体与激励安排，仅靠教师个体自发合作难以持续。

评价落地困难，则源于 AI 生成代码的改写程度难以量化，过程性材料(调试记录、AI 对话记录等)的收集与甄别显著增加了教师工作量，在现有考核框架下执行成本较高。这一问题的本质，是过程性评价的设计初衷(捕捉真实学习过程)与其落地成本(依赖人工逐份核对)之间的张力，单纯靠“增加教师投入”难以持续，亟须借助技术手段加以纾解。

5. 农林类高职院校 AI 辅助编程教学优化路径

针对上述归因，提出与问题一一对应的优化路径，实现“问题–归因–对策”的闭环。

5.1. 制度先行：建立课堂 AI 工具使用规范，构建防依赖约束机制

针对学生过度依赖问题，应在制度层面明确 AI 的辅助定位——AI 是辅助思维的工具，而非替代代码完成的工具，厘清 AI 合理使用与学术不端的边界。具体措施包括：规定学生不得直接提交未经理解与修改的 AI 完整代码；要求学生对 AI 生成代码进行注释、改写与调试；课堂增加口头提问环节，让学生解释代码逻辑。同时建立作业材料留存制度，要求提交 AI 对话记录、代码修改注释与调试过程记录，将代码阅读理解与改写程度纳入过程考核，从评价制度倒逼学生深度思考。

在制度设计上，还应引入“有计划的 AI 淡出(fading)策略”：随着课程推进，逐步收缩 AI 的介入范围——从入门阶段允许 AI 承担报错解释，到中期要求 AI 仅提供提示、学生自主完成代码，再到综合实训阶段设置“关闭 AI”的独立完成环节与阶段测验。通过脚手架的系统性撤除，促使学生把在 AI 支撑下获得的能力逐步内化为独立能力，从源头上规避“元认知卸载”风险[11] [12]。

5.2. 跨域协同：构建智慧农业场景实训资源供给机制

针对农林特色资源不足，应建立软件教师与农业专业教师、行业协会专家联合开发的跨域协同机制。为使这一机制从“倡导”走向“落地”，研究提出“微型项目起步、梯度推进”的具体实施路径：

第一步，以单个教学案例为最小合作单元起步。由软件教师提出明确的编程教学目标与难度约束，农业专业教师提供真实的农业业务场景(如某一种植基地的传感器数据采集需求、某农产品的产销数据)，双方共同完成一个“微型项目”的教学案例开发(如“读取并可视化一地块温湿度数据”)。这种单案例合作门槛低、周期短、易见效，能够在不显著增加教师负担的前提下快速积累第一批可复用的农林案例。

第二步，由点及面，形成案例库与课程群。在单案例合作跑通后，将成功的合作模式复制到更多业务场景，逐步在 6 门课程中嵌入梯度化农林行业小项目——Python 课程扩展农业大数据看板、销售数据可视化分析，Java 课程深化作物价格趋势可视化，移动互联开发课程深化智慧农业移动应用，HTML + CSS 制作农产品展示网站，Linux 部署农业信息服务系统，使农林特色切实落实到课程教学之中。

第三步，依托院系级激励政策保障可持续性。跨专业协同的持续运转不能仅靠教师热情，需要制度化的激励安排：一是将跨专业教学案例开发纳入院系教学工作量认定，对参与协同开发的农业专业教师给予课时折算或绩效倾斜；二是设立面向“智慧农业 + 编程”交叉方向的校级教研课题立项通道，以立项支持形式引导教师持续投入；三是依托学院智慧农业专业群与上海市计算机行业协会资源，为协同开发提供行业专家指导与真实项目来源。通过“工作量认定 + 课题立项 + 行业资源”的组合激励，把跨域协同从临时性合作固化为常态化机制。

5.3. 分层适配：完善“免费 + 付费”组合式工具实施方案

针对经费与网络约束，坚持分层适配原则：日常基础教学以免费、可本地部署的工具为主，降低经费压力；综合实训阶段按需选择付费专业工具，做好成本与效益的权衡。建立工具测评更新机制，定期评估工具在教学环境下的可用性，同步优化机房网络保障，确保课堂实施的稳定性。

5.4. 减负增效：以技术手段优化面向 AI 时代的多元过程评价

针对评价执行成本高的问题，应在保留过程评价精神的前提下，引入技术手段辅助评价，把教师从机械性、重复性的人工核对中解放出来。具体可探索以下三类技术路径：

其一，基于代码版本控制系统的日志分析。3.5 节已要求学生在 Git 等版本控制工具中以分步提交记录代码演进。借助提交历史(commit history)、分支结构与差异日志(diff log)，教师可以低成本地还原学生的开发过程：是“多日迭代、逐步完善”，还是“一次性提交完整成品”。前者反映真实的学习过程，后者则提示可能存在复制粘贴或 AI 代写。Git 的提交时间戳、修改粒度与提交信息，为甄别“过程真实性”提供了客观、可回溯的证据，大幅降低了对 AI 生成作业的甄别成本[13]。

其二，基于自动化脚本的改写度量化。可编写轻量级脚本，对 AI 生成代码与学生提交代码进行比对，自动计算代码改写率、注释覆盖率、变量命名变更等指标，量化学生对 AI 生成代码的理解与改写程度。进一步，可引入静态代码分析工具(如 ESLint、Pylint 等)自动检查代码规范性与复杂度，辅助判断代码质量，减少教师逐行审阅的工作量[13]。

其三，结合代码相似度检测工具辅助甄别。借助代码相似度检测工具(如 JPlag、MOSS 等)对学生提交代码进行两两比对，识别高度相似或疑似 AI 生成的代码片段，作为教师进一步质询的线索，而非直接定性的依据[13]。

需要强调的是，技术手段的作用是“辅助”而非“替代”教师判断。上述工具旨在把教师的精力从机械核对中解放出来，聚焦于最有价值的环节——通过课堂口头答辩、代码逻辑讲解等方式，考查学生

是否真正理解所提交的代码。同时，通过任务单模板化减少教师重复备课工作量，使过程评价在有效防范依赖的同时不显著增加教师负担。

5.5. 师资赋能：建立课程组教研机制

针对教师 AI 教学能力与跨专业背景不足，应建立课程组集体教研机制，围绕与 AI 交互的提问设计、课堂风险管控、AI 生成代码甄别等主题开展研讨，交流不同课程 AI 融入的教学流程。可引入行业协会技术专家参与课程资源打磨与师资培训，提升教师驾驭 AI 编程课堂的能力。同时，结合 5.2 节的跨域协同机制，引导软件教师在与农业专业教师的合作中逐步积累农业行业知识，缓解“懂技术不懂农业”的师资短板。

5.6. 推广启示

农林类高职院校的 AI 辅助编程教学改革，不能直接照搬理工类院校的现成方案，而须立足自身智慧农业人才培养定位进行本土化改造。改革成功的关键，不在于引入何种 AI 工具，而在于是否构建了适配本校学情、办学特色与资源条件的制度、资源与评价体系。本研究的经验可归纳为三点可迁移的启示：其一，理论先行——以认知负荷、自我调节学习等理论为指引，预先识别 AI 的双重效应，把“防依赖”设计进教学流程而非事后补救；其二，评价借力技术——善用版本控制日志、自动化脚本与相似度检测工具，让过程性评价“减负不减质”；其三，协同由小及大——以微型项目起步培育跨专业合作，再以院系级激励政策将其固化。

6. 结论与展望

6.1. 主要研究结论

基于两个学期的教学实践，本研究得出三点结论。第一，“教师引导 + AI 辅助 + 项目实操”模式在农林类高职软件技术专业具备可行性，能够有效降低编程入门难度、提升代码完成效率与学习信心，但仅将 AI 工具引入课堂难以达到理想效果，必须配套制度、资源、评价等系统性改造。第二，AI 赋能是认知脚手架而非编程逻辑训练的替代，其作用具有双重效应：既能通过削减外在认知负荷支持学生入门，也可能诱发认知卸载与工具依赖；学生过度依赖的根源不限于个体，而是学情、评价制度、课堂规则多重因素叠加，需从制度与评价层面、并辅以“有计划淡出”策略加以化解。第三，本研究最核心的经验在于：农林类高职院校改革的关键，不是创建某种普适的新教学模式，而是把通用的 AI 辅助编程方法，在“对接智慧农业人才需求”这一特定目标下进行本土化改造——打通软件技术与智慧农业业务场景、以农林特色项目替代通用项目、以适配本校约束的资源与评价体系支撑改革落地，这既是其改革的价值所在，也是区别于普通院校的核心优势。

6.2. 研究局限

本研究存在一定局限：样本来自本校软件技术专业，向其他院校推广存在局限；实验周期为两个学期，对长期效果的考察尚不充分；智慧农业方向的大型深度项目实训仍显不足，农林特色发挥有待深化；此外，本文对“认知卸载”与“自我调节学习能力弱化”的观察，目前主要基于问卷、访谈与教师经验判断，尚缺乏对元认知过程更为精细的测量工具与纵向追踪数据。

6.3. 未来展望

后续研究可从四方面深化：一是进一步丰富农林场景实训资源，依托跨域协同机制开发更贴近真实农业业务的系统化案例；二是探索 AI 生成作业的智能甄别手段，完善基于版本控制日志与相似度检测的

防依赖技术支撑，并开展其在教学中的有效性验证；三是持续迭代过程性评价指标体系，使其更为科学、更易执行；四是引入更系统的元认知测量与纵向追踪研究，进一步检验“有计划淡出”策略对缓解认知卸载、提升自我调节学习能力的实际效果。农林类高职院校应立足区域农业数字化需求，将 AI 辅助编程教学改革推向纵深，为乡村振兴培养更多复合型数字化技术技能人才。

基金项目

上海农林职业技术学院校内课题：AI 辅助编程工具在高职软件技术专业课程中的应用研究(编号：JY6-0000-26-04)。

参考文献

[1] 孙洪迪, 贾民政, 杨民峰. 基于 ChatGPT 的课程学习助手系统的设计与实现[J]. 北京工业职业技术学院学报, 2024, 23(1): 17-22.

[2] 薄钧戈, 乔亚男, 齐琪, 刘虎军, 黄鑫. 探索 AIGC 技术在高校编程课程中的应用潜力与挑战[J]. 计算机技术与发展, 2024, 34(6): 214-220.

[3] 贾民政, 孙洪迪. 基于AIGC代码助手的高职编程教学应用研究[J]. 北京工业职业技术学院学报, 2025, 24(3): 71-75.

[4] 王新宇. 人工智能赋能高职计算机课程教学改革策略探究[J]. 电脑知识与技术, 2026, 22(3): 161-163.

[5] 杨现民, 郑奕珂, 李新, 黄璐璐. 加速推进人工智能与教育的深度融合: 经验、挑战与建议——基于 8 场专题座谈的内容分析[J]. 中国电化教育, 2025(7): 34-41.

[6] 姜永成, 覃继媛. 基于 AIGC 的高职 Python 程序设计项目式教学设计与实践[J]. 电脑与信息技术, 2026, 34(1): 143-146.

[7] Kazemitabaar, M., Chow, J., Ma, C.K.T., Ericson, B.J., Weintrop, D. and Grossman, T. (2023) Studying the Effect of AI Code Generators on Supporting Novice Learners in Introductory Programming. *Proceedings of the* 2023 *CHI Conference on Human Factors in Computing Systems*, Hamburg, 23-28 April 2023, Article No. 455. https://doi.org/10.1145/3544548.3580919

[8] Sweller, J. (1988) Cognitive Load during Problem Solving: Effects on Learning. *Cognitive Science*, **12**, 257-285. https://doi.org/10.1207/s15516709cog1202_4

[9] Vygotsky, L.S. (1978) Mind in Society: The Development of Higher Psychological Processes. Harvard University Press.

[10] Zimmerman, B.J. (2000) Attaining Self-Regulation: A Social Cognitive Perspective. In: Boekaerts, M., Pintrich, P.R. and Zeidner, M., Eds., *Handbook of Self-Regulation*, Elsevier, 13-39. https://doi.org/10.1016/b978-012109890-2/50031-7

[11] Risko, E.F. and Gilbert, S.J. (2016) Cognitive Offloading. *Trends in Cognitive Sciences*, **20**, 676-688. https://doi.org/10.1016/j.tics.2016.07.002

[12] Liu, D., Fan, G. and Pan, L. (2026) Tool, Tutor, or Crutch? A Grounded Theory of Cognitive Scaffolding and Offloading in AI-Assisted Programming Education. *International Journal of STEM Education*, **13**, Article No. 10. https://doi.org/10.1186/s40594-025-00592-w

[13] Garcia, M.B., Yousef, A.M.F., Yılmaz, R., Sharma, R.C., Chiu, T.K.F. and Damaševičius, R. (2026) Assessment of Coding Skills and Programming Knowledge in the Age of Generative AI: Best Practices and Effective Strategies for Computer Science Education. In: *Pedagogical Innovations in Computer Science Education*, IGI Global Scientific Publishing, 275-304. https://doi.org/10.4018/979-8-3373-6546-6.ch009