ARTIFICIAL INTELLIGENCE: A POWERFUL ASSISTANT IN SELF-LEARNING FOR COMPUTER SCIENCE STUDENTS

LE THI KIM LOAN
Pedagogical Department, Phu Yen University

Received: 15 November 2025; Revised: 29 November 2025; Accepted for publication: 13 December 2025.

Abstract

In the context of higher education strongly transforming itself under the impact of technology, Artificial Intelligence (AI) has become an important support tool in the learning process, especially for Computer Science (CS) students. Integrating a structured literature review and conceptual analysis, this paper evaluates the role of AI as a self-learning assistant, focusing on outstanding benefits such as personalized learning, source code debugging, and explaining complex algorithms, as well as challenges related to academic integrity, ethics, and independent thinking skills of students. Through analysis, the paper proposes a comprehensive recommendation framework, including: (1) an in-depth analysis of the five most effective AI tools; (2) smart strategies for AI-use through Prompt Engineering; (3) new pedagogical approaches for instructors; and (4) support policies from institutions. The results indicate that although AI cannot substitute human instructors completely, it can act as a powerful lever to enhance self-learning capabilities. When applied appropriately, it contributes to elevating the quality of Computer Science education in universities.

Keywords: *AI-assisted learning, Artificial Intelligence, Computer Science students, higher education, self-learning assistant.*

1. Introduction

The field of Computer Science (CS) is characterized by the rapid changing pace of technology. The evolution of frameworks, programming languages, and system architectures means that knowledge acquired in the classroom can quickly become outdated. Consequently, self-learning and lifelong learning no longer remain merely as supplementary skills but have become essential competencies for students in this field. However, students, in their self-learning process, encounter several obstacles: advanced materials are densely written in academic English, debugging source code takes hours without identifying causes, and a lack of one-on-one mentors for immediate support.

On the other hand, AI has been one of the key technologies that has had a profound impact on various fields, including higher education. The emergence of Large Language Models (LLMs), adaptive learning systems, programming assistants, and content creation tools has opened up a new era, innovating students' learning methods. According to UNESCO (2023), AI has the potential to fundamentally change the way learners access knowledge and develop skills, especially in higher education environments where self-learning plays a key role.

While traditional teaching methods rely primarily on rigid lectures, uniform learning pace, and direct guidance from lecturers, students often have difficulty accessing advanced knowledge or support outside of class. In contrast, AI-powered learning tools introduce a more dynamic and adaptive environment, enabling learners to access tailored explanations, varied difficulty levels, and on-demand assistance. Recent studies further highlight that traditional learning models lack flexibility, making them less suitable than AI systems for disciplines that require swift updates, such as CS. LLMs are

Email: leloandhpy@gmail.com

DOI: 10.64410/IBHC3913

capable of providing more diverse explanations and illustrative examples than previous static learning sources (Kasneci et al., 2025; Sharma et al., 2025).

For CS students, AI is not merely a research subject but also a powerful assistant in their self-learning. Thanks to its capabilities in natural language processing, data analysis, and simulation, AI facilitates explaining complex algorithms, debugging code, generating test cases, and proposing viable solutions to technical problems. As observed by Yan et al. (2025) and Massaty et al. (2024), AI has the potential to serve as a "virtual tutor", guiding learners step by step through decision-making processes and fostering the development of computational thinking. However, the integration of AI in self-learning is not without challenges. Several studies have warned that overreliance on smart systems may cause deterioration in independent thinking and risks of *"ability hallucination"* on students' part (Zhai et al., 2024; Becker et al., 2023).

Therefore, it is imperative to examine the role of AI as a self-learning supporter for CS students, identifying how AI can be utilized effectively and appropriately, while ensuring the development of students' professional competencies. By integrating a review of published literature with conceptual analysis, this paper analyzes both the advantages and drawbacks of applying AI in students' self-learning processes and proposes practical directions for its use, with the view to leveraging the potential of emerging technologies while still maintaining and developing independent thinking and specialized skills required in the field. This is a fundamental step to transit from traditional teaching models to AI-augmented education, where humans and machines collaborate to optimize knowledge acquisition.

2. The Role of AI as a Self-Learning Assistant for CS Students

The application of AI in higher education is opening up new opportunities for self-learning, especially for students majoring in CS. In this context, AI is reshaping self-learning methods through these primary roles, transforming passive learning into an active, highly interactive approach.

2.1. Personalized Tutor and Definition Explanation

AI possesses the ability to simplify abstract ideas tailored to each student's ability, thus enabling personalized tutoring. Recent studies have shown that AI tools support personalized learning, simulation of complex knowledge, and significantly improve learning performance (Abrar et al., 2025; Praba & Sanjai, 2025). For example, students can prompt AI for Dynamic Programming explanations at different levels: from beginner-friendly overview to advanced mathematical formulations. The 24/7 availability of AI also allows learners to maintain continuity in their studies without interruptions caused by waiting for instructor feedback (Mollick & Mollick, 2023).

AI systems can analyze individual learning data to generate a customized learning path tailored to each learner's abilities, preventing cognitive overload or knowledge gaps. According to Subhalakshmi et al. (2025), adaptive learning environments driven by AI can increase learning efficiency and engagement by 25-30% by adjusting instructional content to learners' pace and prior knowledge. While traditional teaching models, which impose uniform pacing across an entire cohort, "often struggle to adapt dynamically" to students' demands, AI allows progressing along personalized learning paced, an especially valuable capability when dealing with abstract or mathematically intensive material in CS.

2.2. Intelligent Coding Assistance

AI enhances programming proficiency by offering immediate, context-specific feedback. Cui et al. (2024), through a trial, found that programmers using GitHub Copilot reduced debugging time significantly, increased coding and prototyping speed by 20-30%, and overall completed tasks 55% faster. The rapid feedback loop of AI allows users, including students, to maintain cognitive flow and direct their attention toward problem-solving and software design rather than spending disproportionate time troubleshooting.

2.1.1. Automated test case generation

Testing is a fundamental component of CS education, teaching students how to verify program correctness, anticipate edge cases, and evaluate software robustness. However, a developer needs to possess high-level skills to perform this effectively, and inexperienced programmers often find themselves struggling to design a comprehensive test suite, leading to unexpected bugs and performance issues (Nachiengmai et al., 2020). AI-powered tools address this challenge by automatically generating diverse and structured test cases, enabling students to evaluate the correctness and robustness of their implementations more thoroughly.

Moreover, these systems analyze students' code logic, identifying potential failure points and inputs exposing obscured bugs. For example, an AI tool might detect that a sorting algorithm fails to handle empty arrays or duplicate values, thus proposing targeted tests to reveal these flaws. This process not only ensures the accuracy of debugging but also trains students to think more critically about how each snippet of code behaves under different conditions.

By observing how AI implements professional testing practices (unit testing, integration testing, property-based testing…), students can learn the fundamental principles of code coverage, invariants, and software reliability. This strengthens students' understanding of quality assurance and reinforces the mindset that testing is an indispensable part of modern software development.

2.1.2. Debugging and error explanations

Debugging is a core skill in programming, teaching individuals how to reason about program behavior, identify the root causes of failures, and develop systematic problem-solving habits. Yet, it is also one of the most frustrating aspects of coding. Beginners often find difficulty in interpreting cryptic compiler messages and runtime errors, leading to frustration and discouragement in learning. With the emergence of modern AI tools, this process is now less laborious. AI-powered debugging tools provide clear, context-aware explanations that guide students toward understanding both the cause of error and its optimal solution.

Unlike traditional static analysis, which often provides generic error messages, AI systems can analyze the learner's intent and trace execution flow to identify deeper logical misconceptions. In a scenario involving improper memory management in C++, for example, AI can highlight the exact line where memory is mishandled, explain the difference between stack and heap allocation, and demonstrate safer alternatives such as smart pointers.

Furthermore, AI facilitates the diagnosis of complex runtime behaviors, such as infinite loops or concurrency errors, through step-by-step reasoning traces. These explanations assist students in constructing mentally the program execution, enabling a switch from pure guessing to analytical debugging. By reducing cognitive load and offering immediate, personalized clarification, AI tools allow students to focus on fundamental programming concepts and develop more optimal problem-solving strategies.

3. Code Review and Suggestions

AI-powered suggestion systems now significantly reduce the time students spend refining their code by offering instant, structured feedback that mirrors real-world review processes. Instead of waiting for instructors or peer feedback to revise their work, learners today receive immediate insights into how their code can be improved in readability, structure, and long-term maintainability.

These tools offer multi-dimensional program evaluation, including style consistency, logical flow, modularity, and documentation standards. By identifying problems such as redundancy, overly complex functions, or weak abstractions, they provide targeted guidance on improving code structure. More advanced systems can detect even deeper architectural and safety issues, such as insecure input handling or improper exception management. The feedback exposes students to engineering considerations that are often overlooked in introductory courses, highlighting the importance of robust design principles.

The integration of automated code reviews fosters the ability to write high-quality programs. Receiving consistent feedback on best practices, students will gradually adopt professional standards for maintainability and collaboration early in their studies, progressing from simply writing basic scripts to engineering thoughtful, scalable solutions.

4. Generating Data and Practice Problems

Another key benefit of AI is supporting deeper conceptual understanding through explanation and simulation using its data generating ability. By providing multi-level explanations, visual representations, and interactive simulations, AI helps learners grasp complex ideas more intuitively. Tools based on AI can clarify algorithmic processes, simulate data structures, or illustrate the behavior of neural networks in accessible ways. If students find difficulty in solving an AI-generated problem, they can immediately demand AI to stimulate data processing algorithms, explain procedures at beginner, intermediate, or advanced levels, and even create a diagram illustrating data structures and processes. Students who used AI to support learning in technical subjects demonstrated deeper conceptual understanding and improved ability to articulate knowledge (Guo et al., 2025). As a result, learners switch from passive information receiving to more meaningful internalization, reducing the time required for independent research.

Simultaneously, AI also serves as an intelligent problem generator, capable of producing personalized exercises and rich synthetic datasets for hands-on practice. Instead of static questions, AI offers adaptive difficulty based on personal progress, thus creating a closed learning cycle "trial – error – refinement" tailored to each student. Datasets compiled by AI enable exposure to scenarios resembling the real world, and accordingly, practice becomes not only technical but also conceptual, encouraging students to develop modeling skills, analytical thinking, and problem-solving abilities in intricate environments.

4.1. Supporting Material Comprehension and Scientific Research

A further benefit of AI is to mitigate language barriers in accessing academic materials, which traditionally limit access to global research. Intelligent translation, summarization, and terminology-extraction tools enable students to interpret specialized wording, understand terminology, and condense lengthy materials without losing conceptual coherence. Rather than struggling to search for definitions across multiple sources, students can rely on AI to clarify technical wording, highlight theoretical connections, and provide possible paraphrases aligned with the context of CS. As Shah et al. (2025) highlight, such AI-based reading support greatly expands learners' access to global academic resources, acting as a crucial foundation for newcomers who wish to keep pace with rapidly evolving CS research.

In the context of scientific research itself, AI also functions as a "research assistant" that helps students engage with the research process more systematically. AI can suggest research directions, analyze relevant theoretical frameworks, generate comparison across related studies, or simulate expected outcomes for a proposed research design. In stages involve writing, AI assists in improving structural coherence, reasoning, standardizing citations in formats such as APA or IEEE, and maintaining consistent academic language. Being able to receive instant feedback allows students to gradually develop rigorous research thinking, from understanding the problem to research method design and result interpretation. This is particularly valuable for those who are still unfamiliar with an academic context.

4.2. Learning Path Consulting

Additionally, AI can play a role in guiding students through their learning journeys. Based on a student's career goals (as an AI engineer, web developer, or cybersecurity specialist), AI can generate a highly personalized learning roadmap. This includes recommending foundational textbooks, practical projects, and step-by-step milestones. By analyzing the student's current skill level, learning style, and long-term objectives, AI is able to adjust the difficulty level, suggest alternative resources, and highlight which skills should be prioritized. More importantly, it acts as a close mentor: monitoring progress, identifying gaps, reminding students of upcoming goals, and adapting the roadmap as interests evolve. With such tailored guidance, students can filter excessive information and simply focus on the knowledge that truly contributes to their development.

5. The Limitations of AI for CS Students' Self-learning

Besides its benefits, the use of AI in self- learning is associated with notable risks that CS students need to be fully aware of in order to avoid long-term consequences.

5.1. Over-reliance and Critical Thinking Deterioration

One of the most prominent concerns regarding the integration of AI into education is the risk of students becoming overly dependent on it, which can undermine their ability to think and solve problems independently. Research indicates that excessive use of AI in tasks such as programming and academic writing may lead to over-reliance, where learners passively accept AI-generated outputs without sufficient verification or reflection (Zhai et al., 2024). When answers are delivered instantly, students tend to analyze less deeply, question less frequently, and gradually lose motivation to explore the underlying principles behind solutions. Over time, this pattern can weaken critical thinking and analytical reasoning, which are fundamental competencies in CS education.

This concern becomes particularly serious in programming contexts. Nowadays, when AI tools can generate code quickly and accurately, students may develop a tendency of "cognitive offloading". Instead of handling algorithms and logic to develop problem-solving skills, learners simply copy AI-produced code without fully understanding it. While this may improve short-term productivity, it poses long-term risks. Students may struggle to adapt when faced with novel, complex problems that remain outside the scope of AI training data. As Qadir (2023) states, such dependence can limit learners' innovation and creativity, mitigating their preparedness for real-world challenges.

5.2. Concerns of "Hallucination" and Unreliability

Furthermore, LLMs can generate incorrect information yet presenting it fluently and convincingly, a phenomenon known as "*hallucination*". This is because they operate based on statistical probabilities rather than on a verified knowledge base. AI models sometimes fabricate data, cite nonexistent research, or produce incorrect solutions that "sound plausible" (Kalai et al., 2025; Huang et al., 2024). In programming, AI may generate libraries or API functions that do not actually exist. If students place blind trust and use them without verification, they may waste significant time and develop misconceptions or incorrect knowledge (Ji et al., 2023).

For self-learning students, especially those lacking a solid foundational background, the risk of absorbing and spreading misinformation is extremely high. Indeed, findings have shown that students' information verification skills have not kept pace with the rapid development of AI technologies. Many students tended to trust AI-generated answers more readily than they cross-check with academic sources (Martin-Moncunill & Martinez, 2025). This tendency is especially dangerous in fields requiring high precision, such as algorithms, data analysis, or computer networks. Without strong source evaluation skills and cross-checking with reliable materials, students can be easily misled and face difficulties when pursuing advanced self-study.

5.3. Academic Integrity, Ethics, and Security

The widespread integration of AI in higher education presents substantial challenges to academic integrity. As AI now supports more advanced functions, such as generating essays, solving assignments, and producing source code, the boundary between assistance and academic misconduct becomes more blurred than ever. When students present AI-generated work as their own, it is actually very difficult to identify the actual author and assess whether learning objectives have been met. This concern is exacerbated by the fact that AI detection tools remain underdeveloped and frequently unreliable, making it challenging for educators to verify the authenticity of student submissions (Farrelly & Baker, 2023). Adopting such practices without sufficient understanding not only compromises learning outcomes in the long term but also risks violating institutional policies on originality and authorship.

Beyond integrity, ethical issues arise regarding responsibility, fairness, and authorship in AI-assisted learning. Generative AI models are trained on large-scale datasets that include existing texts, code, and other intellectual works, raising concerns about implicit reproduction of ideas without clear attribution (Lucchi, 2023). Using AI-generated content, students may unknowingly reproduce results derived from prior sources, blurring ethical boundaries around originality and ownership. Additionally, when students utilize AI as a convenient "ghostwriter", they misrepresent the effort invested and the level of mastery achieved, creating a misleading picture of their skills. This negatively fosters a sense of irresponsibility in the students, which may further undermine the ethical nature of education, which is originally based on trust and personal accountability.

The use of AI tools also introduces important security and privacy vulnerabilities. Students may inadvertently input sensitive information, such as API keys, database credentials, or proprietary source code from internship or research projects when interacting with public AI platforms, leading to information leakage and potential legal or ethical violations. At the same time, it is hard to trace AI-generated content back to its original sources, which complicates plagiarism detection and data governance. These vulnerabilities allow proprietary or sensitive information to circulate without effective control, presenting a limitation in modern educational environments where academic purposes are taken advantage of to violate data privacy and information security.

5.4. Access Inequality

Moreover, unequal access to AI tools poses a significant challenge to fairness in AI-integrated learning. Differences in device quality, internet bandwidth, English proficiency, or technological experience critically influence how well students can utilize AI systems. Instead of bridging educational gaps, these disparities may generate new forms of educational disparity, further accentuating existing socio-economic and digital divides. Such inequality in access is directly correlated with diminished learning productivity and weaker academic outcomes (Ahmed, 2024). To mitigate this challenge, it is imperative that universities play a role in providing necessary AI literacy training, ensuring AI adoption enhances educational equity rather than widening existing skill gaps.

In short, although AI provides significant benefits for students in CS, its use also introduces various risks and challenges. These issues are not only related to the reliability of AI tools but also affect students' cognitive abilities, academic integrity, and accessibility to technology. Therefore, AI should be approached with caution and critical awareness, rather than being viewed as a perfect solution that can fully replace traditional learning methods.

6. Recommendation for Effective AI Use As a Self-learning Assistant

To transform AI into an effective tool rather than a "double-edged sword," a coordinated approach is required, encompassing careful tool selection as well as fundamental changes in teaching and learning methodologies.

6.1. The Five Most Effective AI Tools for CS Students

Table 1. The five most effective AI tools for CS students

Tool	Primary role	Key features	Optimal use scenarios
1. ChatGPT	Versatile tutor	Socratic Method capabilities, multi-level explanations, and broad knowledge base for theoretical concepts.	Learning new algorithms, mocking technical interviews, and generating practice problems for exams.
2. GitHub Copilot	Paired programmer	IDE integration, context-aware code suggestions, and conversion of natural language comments into code.	Writing boilerplate code, accelerating typing speed, and real-time syntax error correction.
3. Claude 3.5 Sonnet	Logical expert & reviewer	Large output limit strong reasoning capabilities, and "Artifacts" feature for visualizing code/UI side-by-side.	Code review, refactoring long files, debugging complex logic, and generating system architecture diagrams.

Tool	Primary role	Key features	Optimal use scenarios
4. Google AI Studio	Deep context analyzer	1M+ token context window, multimodal ingestion (PDFs, long videos, folders), and adjustable system instructions.	Uploading full textbooks or lecture recordings for Q&A, querying specific details across massive documentation, and learning prompt engineering.
5. Cursor	AI-native code editor	Codebase indexing, ability to edit multiple files simultaneously ("Composer" mode), and built-in terminal integration.	Building full-stack features across multiple files, navigating unfamiliar open-source projects, and seamless debugging.

6.1.1. ChatGPT

Usage strategy: Simulated assessment & Active recall. Instead of passively reading notes, students can utilize ChatGPT to simulate exams or job interviews to identify knowledge gaps.

Specific example: In a Data Structures course, students can prompt ChatGPT for a sample quiz on the "Stack" concept (LIFO) and immediate feedback to reinforce understanding.

6.1.2. GitHub Copilot

Usage strategy: Comment-Driven Development (CDD). Students should write comments describing the logic before typing the code.

Specific example: When implementing a recursive factorial function in Python, the student should type a clear comment: *# Recursive function to calculate the factorial of n* and observe how Copilot translates clear comments into code. The student must review the code and ensure understanding of the logic before accepting the code.

6.1.3. Claude 3.5 Sonnet

Usage strategy: Deep code analysis & refactoring. Claude is best used to assess system architecture or optimize the performance of an entire project, rather than a single function.

Specific example: A student completes a Web API project consisting of five main source files. Claude can assist in analyzing the project using principles such as DRY, KISS, and YAGNI, and suggest refactoring strategies to improve maintainability.

6.1.4. Google AI Studio

Usage strategy: Multimodal knowledge extraction. Students should utilize the massive context window (1M+ tokens) to process heavy learning materials like full textbooks or lecture videos.

Specific example: A student has a PDF textbook chapter on "Object-Oriented Programming" and is struggling to understand it. Instead of searching for generic definitions, the student uploads the specific PDF file onto Google AI Studio and instantly receives simplified, context-specific explanations of concepts like polymorphism.

6.1.5.Cursor

Usage strategy: Cross-file contextual editing. Unlike standard extensions, students should use Cursor's "Composer" mode to implement features that require changing multiple files at once.

Specific example: A student is building a simple "To-Do List" website and they want to add a "Delete" button to the tasks. Cursor's Composer mode can be used to deploy this by simultaneously updating the HTML structure and JavaScript logic.

6.2. General Guidelines for Appropriate Use of AI

To ensure that AI contributes positively to self-learning, users need to have a clear understanding of how to use these tools responsibly. AI should be regarded as an assistant supporting learning activities rather than a substitute for the learner's own reasoning. In this context, effective prompt engineering plays a critical role, as the quality of AI-generated responses is highly dependent on the quality of user input. Providing clear contextual information, explicitly assigning a role to the AI (reviewer or explainer), and specifying the desired output format can encourage more analytical and educationally meaningful responses.

Additionally, individuals need to verify the accuracy of AI-generated information by comparing it with textbooks, and reputable academic materials. When interacting with AI systems, users are encouraged to pose follow-up analytical and explanatory questions, such as asking for step-by-step reasoning or example-based clarifications, to deepen understanding instead of relying on direct answers. Maintaining an active role in the learning process is essential; AI may offer initial guidance, but the learner must ultimately develop, implement, and evaluate solutions independently.

In parallel, issues of privacy and data protection must be taken seriously, particularly in fields such as CS, where project data may contain sensitive or proprietary content. By adhering to these practices, individuals can benefit from AI's strengths while preserving the development of independent thinking and reducing exposure to misinformation.

6.3. Recommendations for CS students

For CS students, effective integration of AI requires balancing the advantages of technological tools with the foundation of traditional learning methods. Engaging with textbooks, working through problems manually, and discussing ideas with peers provide essential conceptual grounding. Within this context, AI can be used to model, test, and refine ideas prior to full implementation, thereby enhancing students' ability to design and evaluate computational solutions. Students must absolutely avoid using AI as a blind debugging mechanism that simply replaces error code without understanding. Students should always ask themselves and the AI tools why the original code is faulty and what underlying principles cause the error, in order to develop reasoning skills from mistakes.

Another important aspect is cultivating information-evaluation skills, as students must consistently assess the credibility of AI outputs, identify potential logical inconsistencies, and verify claims against reliable sources. Through these practices, students gradually learn to position AI as a complementary tool that supports, rather than dominates their learning process.

6.4. Recommendations for Institutions and Instructors

Educational institutions and instructors play an essential role in guiding the responsible use of AI in academic settings. Clear policies on permissible AI use are necessary to prevent academic misconduct and maintain fairness. At the curricular level, programs should integrate digital literacy and information-verification skills, particularly in courses involving programming, data analysis, and research methods. Instructors are encouraged to create learning environments where AI serves as a supportive tool, yet students are still required to demonstrate their own analytical reasoning in assessments and project work. Finally, institutions should ensure that students have equitable access to technological resources and relevant training, helping to narrow gaps in AI adoption across the student population.

7. Conclusion

In summary, AI is becoming an increasingly influential component of higher education, offering substantial benefits for students, particularly those in CS, through personalized learning, complex knowledge simulation, and adaptive programming support. These not only enhance the flexibility and effectiveness of self-learning but also reshape how students engage with course content and develop core competencies. However, the rapid integration of AI tools also highlights notable challenges, including the spread of inaccurate information, technological overreliance, concerns over academic integrity, and persistent inequalities in access to digital resources. Addressing these issues requires a balanced approach that recognizes both the transformative potential and the inherent limitations of AI within the learning ecosystem.

Looking forward, the effective use of AI in universities depends on the collaborative efforts of students, educators, and institutions. Students must strengthen critical-thinking and information-evaluation skills to navigate AI-generated content responsibly, while instructors and administrators should establish clear guidelines, ethical frameworks, and supportive learning environments that integrate AI without compromising academic standards. Future research should continue to examine the long-term impact of AI on learning behaviors, develop more optimal adaptive learning models, and explore hybrid instructional approaches that combine human expertise with AI-enhanced tools. Such efforts will be essential for leveraging AI to enrich educational experiences and promote equitable, responsible, and sustainable learning outcomes.

References

Abrar, M., Aboraya, W., Abdulghafor, R., Subramanian, K. P., Husaini, Y. A., & Al Husaini, M. A. (2025). *AI-powered learning pathways: Personalized learning and dynamic assessments.* International Journal of Advanced Computer Science and Applications, 16(1). https://doi.org/10.14569/IJACSA.2025.0160145.

Ahmed, F. (2024). *The digital divide and AI in education: Addressing equity and accessibility.* AI EDIFY Journal, 1(2).

Becker, B. A., Denny, P., Finnie-Ansley, J., Luxton-Reilly, A., Prather, J., & Santos, E. A. (2023). *Programming is hard-or at least it used to be: Educational opportunities and challenges of AI code generation.* In Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (pp. 500–506).

Cui, K. Z., Demirer, M., Jaffe, S., Musolff, L., Peng, S., & Salz, T. (2024). *The productivity effects of generative AI: Evidence from a field experiment with GitHub Copilot. An MIT Exploration of Generative AI.* https://doi.org/10.21428/e4baedd9.3ad85f1c.

Farrelly, T., & Baker, N. (2023). *Generative artificial intelligence: Implications and considerations for higher education practice.* Education Sciences, 13(11), 1109.

Guo, Q., Zhen, J., Wu, F., He, Y., & Qiao, C. (2025). *Can Students Make STEM Progress With the Large Language Models (LLMs)? An Empirical Study of LLMs Integration Within Middle School Science and Engineering Practice.* Journal of Educational Computing Research, 63(2), 372–405.

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., & Liu, T. (2025). *A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions.* ACM Transactions on Information Systems, 43(2), 1–55. https://doi.org/10.1145/3703155.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., … & Fung, P. (2023). *Survey of Hallucination in Natural Language Generation.* ACM Computing Surveys, 55(12), 1–38. https://doi.org/10.1145/3571730.

Kalai, A. T., Nachum, O., Vempala, S. S., & Zhang, E. (2025). *Why language models hallucinate.* arXiv preprint. https://doi.org/10.48550/arXiv.2509.04664.

Kasneci, E., Sessler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh, G., Günnemann, S., Hüllermeier, E., Krusche, S., Kutyniok, G., Michaeli, T., Nerdel, C., Pfeffer, J., Poquet, O., Sailer, M., Schmidt, A., Seidel, T., … Kasneci, G. (2023). *ChatGPT for good? On opportunities and challenges of large language models for education.* Learning and Individual Differences, 103, 102274. https://doi.org/10.1016/j.lindif.2023.102274

Lucchi, N. (2023). *ChatGPT: A Case Study on Copyright Challenges for Generative Artificial Intelligence Systems.* European Journal of Risk Regulation, 15, 602–624. https://doi.org/10.1017/err.2023.59.

Martin-Moncunill, D., & Martinez, A. D. (2025). *Students' Trust in AI and Their Verification Strategies: A Case Study at Camilo José Cela University.* Education Sciences, 15(10), 1307. https://doi.org/10.3390/educsci15101307.

Massaty, M. H., Fahrurozi, S. K., & Budiyanto, C. W. (2024). *The role of AI in fostering computational thinking and self-efficacy in educational settings: A systematic review.* Indonesian Journal of Informatics Education, 8(1), 49–61. https://doi.org/10.20961/ijie.v8i1.89596.

Mollick, E., & Mollick, L. (2023). *Assigning AI: Seven Approaches for Students, with Prompts.* https://doi.org/10.48550/arXiv.2306.10052.

Nachiengmai, W., Ramingwong, S., & Kongkeaw, A. (2020). *Implementing DDD for Automatic Test Case Generation.* International Journal of Information and Education Technology, 10, 117–121. https://doi.org/10.18178/ijiet.2020.10.2.1349.

Qadir, J. (2023). *Engineering Education in the Era of ChatGPT: Promise and Pitfalls of Generative AI for Education.* In 2023 IEEE global engineering education conference (EDUCON) (pp. 1–9). IEEE.

Praba, R., & Sanjai, S. (2025). *AI-Powered Interactive Learning Platforms for Modern Education*. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 11, 1687–1692. https://doi.org/10.32628/CSEIT25112527.

Shah, S. H. R., Altaf, A. R., & Mughal, K. H. (2025). *Exploring the impact of AI tools on reading proficiency among undergraduate students.* The Critical Review of Social Sciences Studies, 3(3), 663–670. https://doi.org/10.59075/gpsyvt20.

Sharma, S., Mittal, P., Kumar, M., & Bhardwaj, V. (2025). *The role of large language models in personalized learning: a systematic review of educational impact.* Discover Sustainability, 6(1), 1–24. https://doi.org/10.1007/s43621-025-01094-z.

Subhalakshmi, R. T., Geetha, S., Dhanabal, S., & Balakrishnan, M. (2025). *ALPOA: Adaptive Learning Path Optimization Algorithm for Personalized E-Learning Experiences.* International Journal of Computational and Experimental Science and Engineering, 11(1). https://doi.org/10.22399/ijcesen.910.

UNESCO (2023). *Global Education Monitoring Report 2023: Technology in education – A tool on whose terms?* Paris, UNESCO.

Yan, Y. M., Chen, C. Q., Hu, Y. B., & Ye, X. D. (2025). *LLM-based collaborative programming: impact on students' computational thinking and self-efficacy.* Humanities and Social Sciences Communications, 12(1), 1–12. https://doi.org/10.1057/s41599-025-04471-1.

Zhai, C., Wibowo, S., & Li, L. D. (2024). *The effects of over-reliance on AI dialogue systems on students' cognitive abilities: a systematic review.* Smart Learning Environments, 11, 28. https://doi.org/10.1186/s40561-024-00316-7.