Integrating AI Into Engineering Education: Practical Use Cases for Teaching and Learning

Rose Gomar, Carleton University, Canada

The European Conference on Education 2025
Official Conference Proceedings

Abstract

As Artificial Intelligence (AI) continues to transform industries, its integration into higher education offers new opportunities to enhance teaching and learning. This paper presents practical use cases of AI in undergraduate computer engineering courses, focusing on how AI can support instructors and improve students' learning experience in a rapidly evolving technological environment. My experience using AI tools to streamline and enrich various aspects of course design and delivery are shared. These include generating quiz and exam questions from lecture notes, creating multiple variations of assessment items, drafting lab manual templates, designing analytical rubrics for lab evaluations, and developing lesson plans and motivational lecture slides. I also used AI to analyze open-ended responses from anonymous mid-semester surveys to identify common themes for a timely and responsive course adjustments. Additionally, to support students' AI literacy and readiness for industry practices, I designed AI-integrated assignments where students used AI tools to develop codes for a given digital design problem, simulate the outcomes, and critically reflect on the AI-generated solutions. Across all these applications, using well-crafted prompts and moderator strategies significantly improved the quality and efficiency of AI outputs. This paper highlights how AI can be effectively incorporated to enhance instructional design, raise student engagement, and support a more responsive and inclusive approach to teaching.

Keywords: artificial intelligence (AI), computer engineering, ChatGPT

Introduction

AI has rapidly emerged as a transformative force across industries, reshaping processes, decision-making, and innovation. In higher education, AI's integration has generated both enthusiasm and concern, prompting scholars and practitioners to explore its potential to enhance teaching, learning, and assessment (Groothuijsen et al., 2024; Qadir, 2023; Walter 2024; Vierhauser et al., 2024). Engineering education, with its emphasis on problem-solving, design, and hands-on experience, stands to benefit significantly from AI tools that can provide personalized learning, automate assessment, and support interactive and experiential learning environments (Liu et al., 2025; Walter, 2024; Yelamarthi et al., 2024).

The release of large language models such as ChatGPT in late 2022 has accelerated this discourse by making advanced natural language generation widely accessible (Liu et al., 2025, Neumann et al., 2023). These models can generate code, summarize literature, create instructional materials, and facilitate formative assessment, offering instructors novel ways to streamline preparation and engage students (Yelamarthi et al., 2024). However, their adoption has also raised pedagogical and ethical questions, including risks of over-reliance, internalizing mistakes by students due to unsupervised use of AI, academic integrity concerns, privacy issues and the need to develop guidelines that define acceptable use in educational contexts (Daun et al., 2023; Liu et al., 2025; Neumann et al., 2023; Vierhauser et al., 2024).

Recent literature on AI in programming education further emphasizes that students' outcomes depend heavily on their ability to critically evaluate and refine AI outputs (Choudhuri et al., 2025; Neumann et al., 2023; Qadir, 2023). Moreover, research highlights the importance of aligning AI integration with course level and learning objectives, as early-stage learners may lack the foundational knowledge required to assess AI-generated solutions effectively (Choudhuri et al., 2025).

While the literature has begun to map AI's capabilities and implications, there remains a notable gap in how these tools can be systematically integrated into instructional tasks—such as lesson planning, activity design, formative feedback, and curriculum development—particularly in engineering contexts. Most studies focus on theoretical affordances or isolated use cases, leaving limited guidance on practical, course-level strategies that address both pedagogical opportunities and challenges.

In this context, the present paper offers a practitioner's perspective on the practical integration of AI into undergraduate computer engineering courses. Drawing from direct teaching experience in large-enrollment courses with substantial laboratory components, it examines specific use cases—including AI-assisted question generation, rubric design, survey analysis, and AI-integrated assignments—that illustrate both the promise and pitfalls of these tools. The goal is to provide actionable insights into how AI can be purposefully embedded into engineering education to enhance instructional design, increase student engagement, and support a more inclusive and responsive learning environment, while maintaining academic rigor and fostering critical AI literacy.

The paper is structured as follows. Next section outlines the teaching context in which these practices have been implemented, including course characteristics, class sizes, and instructional formats. The method section details the AI-enhanced teaching practices adopted, while next section presents concrete use cases that demonstrate the application of AI in lesson planning, assessment design, student engagement, and reflective learning activities. Finally, last section offers concluding remarks and recommendations for effectively and responsibly integrating AI into engineering education.

The Context

The practices described in this section have been implemented in the teaching of computer systems engineering courses at Carleton University, Ottawa, Canada. Specifically, these methods have been applied in Digital Systems Design (SYSC2310, second-year course), Computer Systems Design (SYSC3320, third-year course), and Computer Systems Architecture (SYSC4310, fourth-year course). These are relatively large courses, with over 100 students enrolled in SYSC2310, and approximately 80 students in each of SYSC3320 and SYSC4310. All three courses include experiential learning components, primarily delivered through bi-weekly lab sessions lasting two and a half hours. Given the large enrollment numbers, it is essential to develop engaging instructional materials and provide ample opportunities for self-assessment. In the following section, I describe how I plan to enhance my current teaching practices in these courses. Notably, I have mostly utilized the paid version of ChatGPT as a support tool in this process unless otherwise is specified.

Methods

In this section, I describe my experience using generative AI, specifically ChatGPT, to enhance the effectiveness of my teaching practices. These practices span several core instructional activities in my computer systems engineering courses and include:

- Creating lab manual templates
- Generating quiz questions and variations for real-time or self-paced feedback and assessment
- Designing motivational slides
- Developing ice-breaking activities
- Leveraging learning analytics
- Interpreting mid-semester teaching evaluation surveys
- Designing AI-integrated assignments to explore both the potential and limitations of AI
- Creating experiments to illustrate AI limitations
- Generating and refining rubrics for lab experiments

Based on my experience, several key factors influence the quality and relevance of AI-generated outputs:

- **Context matters**: ChatGPT tends to make assumptions when insufficient context is provided. These assumptions can lead to irrelevant or overly generic content.
- **Logged-in usage improves relevance**: The responses differ noticeably when logged into a personal account versus using ChatGPT anonymously. When logged in, the model learns from prior prompts, resulting in responses that are more aligned with one's teaching context and subject area.
- **Prompt quality is critical**: Providing detailed prompts—such as background information, course level, expected output format, and number of items—leads to significantly more accurate and useful results.
- **Prompt modifiers enhance interaction**: Although ChatGPT does not ask clarifying questions by default, using explicit instructions like “don’t make assumptions” or “ask me three questions before generating a response” can guide it toward better outcomes.

- **Project workspaces improve consistency**: For topics that require multiple iterations, creating a dedicated "project" in ChatGPT allows for persistent instructions and uploaded reference materials, improving coherence across interactions.

Use Cases in Teaching and Learning

Lesson Plan Creation

One common challenge in developing new lectures is creating a coherent and engaging sequence of topics that incorporates examples and active learning opportunities. Generative AI can be highly effective in drafting initial lesson plans. However, outcomes vary significantly based on prompt specificity. For instance, when I asked ChatGPT to "generate a lesson plan to teach finite state machines" without logging in, the resulting plan targeted senior high school or early college audiences. It included examples such as pattern recognition and vending machines—scenarios less relevant for a university-level Digital Systems Design course. In contrast, when I asked the same question while logged into my account, ChatGPT leveraged my prompt history and assumed an appropriate context: an undergraduate digital design course with prerequisites like logic gates, combinational circuits, and basic Verilog. The revised plan included more relevant tools and platforms such as Logisim, EDA Playground, and FPGA boards, significantly improving the usability of the output.

Quiz Generation and Variations for Feedback and Assessment

In large-enrollment courses, manual assessment is resource-intensive, especially when teaching assistant support is limited. To address this, I use tools such as PollEverywhere and Wooclap to deliver real-time or asynchronous self-assessment quizzes. These serve as formative feedback mechanisms that benefit both students and instructors. ChatGPT can assist in generating quiz questions based on lecture content. However, in my experience, generic uploads often yield overly simplistic questions. By specifying the number, format, level of difficulty, and topic focus in the prompt, the generated questions become much more aligned with course expectations. ChatGPT is also useful for producing question variations—though specifying the desired degree of similarity is essential to avoid redundant or overly repetitive items. Careful review remains necessary, as the model can sometimes generate misleading or incorrect combinations of technical concepts (i.e., hallucinations).

Motivational Slides

The first slide of a lecture plays a critical role in capturing students' attention and setting the tone for the session. Connecting course content to real-world applications increases engagement. For example, although SYSC2310 course is generally well-received, the early portion of the course—focusing on Boolean algebra and number representation—can feel abstract and math-heavy. In one lecture on coding schemes (e.g., fixed-point, floating-point, Gray code, and ASCII), I used ChatGPT to generate a motivational slide. It suggested: "Have you ever wondered how your phone stores a text message, or how a calculator displays numbers? That's all possible thanks to codes—structured ways of representing data digitally."

This introductory framing proved highly effective in sparking curiosity and contextualizing the topic, making the lecture more relatable and engaging for students.

Developing Ice-Breaking Activities

One of the effective ways to attract students to the course and build a community is developing ice-breaking activities. With the help of ChatGPT, I could generate a series of interactive question to bring all students to the topic and build an interesting and curios atmosphere. For example, When I asked ChaTGPT to generate ice-breaking questions for my SYSC2310 course, it generated these questions:

- What comes to your mind when you hear "Digital design"?
- Which of these have you used or heard of? Logic Gates, Truth Table, Verilog, Karneough map, FPGA, Boolean Functions
- How many transistors are in a modern CPU in an average PC?

While I previously used ice-breaking activities, I found the ones generated by the ChatGPT much more effective in capturing interest and curiosity, and participation.

Analyzing Teaching-Related Surveys

In order to understand students' learning experience, anonymous pre-course and mid-semester survey is very helpful to get an insight into students' expectations and experience. In pre-course survey, students are often asked about their preferred learning style or anything that can be considered by their instructors to help them succeed in the course. In a mid-semester survey on the other hand, questions normally includes a number of likert questions that measures the effectiveness of different methods or component of the course and two open ended questions that asks "what the students like" and "rooms for improvements". AI can help categorize these comments and prepare an action plan for the rest of the semester or future delivery. This is very helpful to improve the course quality for the students. For this task I used the university provided Co-pilot AI tool that is more secure and does not use or store data for training the models. This is especially helpful for large classes. As an example, an analysis of students' comments from the pre-course survey in SYSC3320 revealed several recurring themes, extracted and grouped using the AI tool Copilot. Many students emphasized the importance of engaging content supported by practical examples and requested that all lecture slides and notes be made available on Brightspace. Others expressed interest in interactive elements such as Wooclap quizzes and suggested incorporating a mix of teaching methods (e.g., chalkboard, whiteboard, iPad) to maintain engagement. Several respondents requested ample practice problems and sample exams to support their preparation, while a few noted accessibility considerations, such as adjusting color schemes for students with color blindness. These insights provide valuable guidance for refining course delivery to better meet students' needs.

Designing AI-Integrated Assignments and Exploring the Potential and Limitations of AI

As AI continues to expand across various sectors, including industry, it is essential that students develop a nuanced understanding of both its potential and its limitations. However, the impact of AI tools on student learning heavily depends on when students use them and how they engage with them.

A recent study (Choudhuri et al., 2025) investigated four phases of AI use in teaching software engineering:

- **L1 (Initial Learning):** Learning new SE concepts from scratch
- **L2 (Incremental Learning):** Deepening or refining existing knowledge

- **I1 (Initial Implementation):** Starting software projects and setting up frameworks
- **I2 (Advanced Implementation):** Optimizing or integrating complex systems

The study found that students perceived AI to be most beneficial during the L1 and I1 phases. Specifically, students highlighted advantages such as personalized support and concept clarification, structured outlines and roadmaps, help with syntax and practice problems, and assistance with brainstorming and initial code scaffolding.

In contrast, students encountered more challenges during the L2 and I2 phases. These included (Choudhuri et al., 2025):

- Misunderstanding generative AI's limitations (C1)
- Difficulty articulating their needs effectively (C2)
- Misalignment with their learning styles or workflows (C3)
- Lack of rationale behind AI-generated outputs (C4)
- Difficulty verifying or adapting AI-generated content (C5)

The root causes of these challenges were identified as (Choudhuri et al., 2025):

- Flawed reasoning (e.g., logical errors, contradictions)
- Poor response quality (e.g., incomplete or inconsistent answers)
- Deceptive outputs (e.g., hallucinations, confirmation bias)
- Context neglect (e.g., failure to consider user preferences or constraints)

In light of this, I have been intentional in how I integrate AI into the computer systems engineering curriculum. For example, I do not recommend the use of AI tools in SYSC2310, a second-year course, as this is a course where lots of initial learning about digital design takes place and students at this stage may lack the foundational knowledge required to critically assess AI-generated content and are more vulnerable to misinformation. However, in upper-year courses such as SYSC3320, I see value in incorporating AI in a structured and reflective way.

One such example is an assignment in SYSC3320, where students are tasked with designing a traffic light controller using AI tools. In addition to completing the design task, students are required to:

- Document the exact prompts they used
- Reflect on their experience using AI

Student reflections varied significantly based on the tools and prompt styles they employed, providing valuable insight into their understanding of both AI's capabilities and its shortcomings. In the SYSC3320 assignment on designing a traffic light controller using AI tools, student reflections revealed a nuanced understanding of AI's capabilities and constraints. Many students appreciated AI's ability to scaffold initial designs, generate structured templates, and clarify Verilog syntax—particularly during the early phases of learning and implementation (L1 and I1). They noted that AI accelerated their workflow, introduced them to new constructs like localparam and case statements, and provided helpful debugging support. However, students also encountered significant limitations, especially during refinement and simulation (L2 and I2). Common issues included syntax errors due to unsupported constructs (e.g., SystemVerilog in Verilog-2001), overcomplicated logic, and a lack of contextual awareness in the generated code. Several students expressed that while AI could produce functional code with the right prompts, it often failed to meet nuanced design requirements without substantial human intervention. Others found that over-reliance on AI hindered their deeper understanding of state machine design and debugging. Overall, students recognized AI as a valuable assistant for learning, but emphasized the importance of critical thinking, clear prompting, and manual validation to ensure correctness and learning efficacy.

Creating Experiments to Illustrate AI Limitations

While research (Choudhuri et al., 2025) suggests that using AI during the early stages of learning—such as in a course like SYSC2310—may not be beneficial, students may not immediately accept this advice. Therefore, transparency about the reasoning behind this guidance is essential. In future course deliveries, I plan to discuss with students the potential drawbacks and risks of over-relying on AI during the foundational stages of learning, drawing on findings from recent studies.

To make these points more tangible, I have designed hands-on experiments that illustrate situations where AI tools are ineffective or misleading. For instance, I have observed that AI often struggles with simplifying Karnaugh maps (K-maps) for Boolean functions, a topic in SYSC2310, frequently producing suboptimal or non-minimal expressions. I will use this as a classroom demonstration to highlight the limitations of AI. Similarly, AI-generated solutions for finite state machine (FSM) design often include incomplete or incorrect state transitions. This, too, will serve as an illustrative example to help students critically evaluate AI-generated content.

Generating and Refining Rubrics for Lab Experiments

The use of analytic rubrics with detailed evaluation criteria is often recommended to enhance transparency and consistency in assessment (Derahman et al., 2024). However, developing such rubrics can be time-consuming and challenging, particularly for instructors managing large classes where grading efficiency is crucial. To address this, I have leveraged AI tools to design and refine rubrics for grading lab reports. These AI-assisted rubrics are then integrated into the learning management system (LMS), enabling teaching assistants to apply them directly during the evaluation process. This approach not only streamlines grading but also promotes fairness and consistency across sections and markers.

An example of a rubric co-developed with AI is shown in Figure 1. This rubric helped align assessment criteria with learning outcomes while reducing ambiguity in expectations and feedback.

Figure 1
Sample Rubric Designed for One of SYSC4310 Lab Experiments

Criteria	Level 4	Level 3	Level 2	Level 1	Criterion Score
Task 1: Verilog code for cache module	15 points Verilog code is included, its correct and is explained briefly.	10 points Verilog code has minor issues or not explained.	7 points Verilog code has major issues, and/or not explained.	0 points missing this component.	/ 15
Task 2: Top-level design Verilog code	15 points Verilog code is included for the top-level design, its correct and is explained briefly.	10 points Verilog code has minor issues or not explained.	7 points Verilog code has major issues, and/or not explained.	0 points missing this component.	/ 15
Task 4: Simulation	30 points Simulation is complete for all 6 test cases and correct waveform is included, and properly explained.	22 points Simulation has minor issues, or missing some cases, or waveforms are not explained or not clear.	14 points Simulations have major issues.	0 points missing simulation.	/ 30

Criteria	Level 4	Level 3	Level 2	Level 1	Criterion Score
Documentations	10 points Well-organized documentation, includes a conclusion, Names and student ID are included in all codes. Codes are submitted as a separate zip file on Brightspace. waveforms are annotated when needed.	7 points No so well-organized report or missing student name and ID in codes, or not following the submission formats.	4 points Major documentation issues (see level 4)	0 points unacceptable report based on elements outlined in level 4.	/ 10
Checkout in the lab and answering TAs Questions.	10 points checked out and answered any questions if there was any.	7 points not all questions were answered.	4 points most questions were not answered.	0 points did not check out.	/ 10

Total	/ 80

Conclusion

This paper has demonstrated that generative AI tools, when purposefully integrated into the teaching of undergraduate computer engineering courses, can provide substantial benefits for both instructors and students. By streamlining tasks such as lesson plan creation, quiz and exam question generation, rubric development, and survey analysis, AI can reduce administrative workload and allow instructors to focus more on pedagogy and student engagement. Moreover, structured AI-integrated assignments and classroom activities can enhance students' AI literacy, encouraging them to engage critically with AI outputs, identify errors, and reflect on the technology's capabilities and limitations.

At the same time, the findings reinforce that successful integration requires careful consideration of course level, learning objectives, and students' readiness to work with AI. As shown in both the literature and classroom experience, early-stage learners may be more vulnerable to AI-generated inaccuracies and over-reliance, whereas upper-year students can benefit from structured opportunities to explore AI tools in more complex and reflective ways. The strategic use of prompt engineering, clear guidance, and instructor oversight is therefore essential to maximize the educational value of AI while mitigating risks such as misinformation, flawed reasoning, and reduced independent problem-solving skills.

Looking ahead, the adoption of AI in engineering education should move beyond ad hoc experimentation toward evidence-informed strategies that align with ethical, pedagogical, and disciplinary standards. This includes developing institutional guidelines for AI use in coursework, training instructors and students in critical AI literacy, and building assessment frameworks that value process and understanding alongside final outputs. By combining the efficiency and creative potential of AI with sound instructional design, educators can foster learning environments that are more engaging, inclusive, and responsive—preparing students not only to use AI effectively but also to question, improve, and innovate with it in their future professional practice.

Declaration of Generative AI and AI-Assisted Technologies in the Writing Process

The author declares that ChatGPT, was used for proofreading and refining the language used in the manuscript.

References

Choudhuri, R., Ramakrishnan, A., Chatterjee, A., Trinkenreich, B., Steinmacher, I., Gerosa, M., & Sarma, A. (2025, April). Insights from the frontline: Genai utilization among software engineering students. In 2025 IEEE/ACM 37th International Conference on Software Engineering Education and Training (CSEE&T) (pp. 1-12). IEEE.

Daun, M., & Brings, J. (2023, June). How ChatGPT will change software engineering education. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1* (pp. 110-116).

Derahman, A., Yusof, M., & Hamzah, N. (2024). Analytic rubrics for the practical assessment of students' performances in engineering survey course. *Journal of Sustainable Civil Engineering & Technology (JSCET)*, *3*(2), 1-10.

Groothuijsen, S., van den Beemt, A., Remmers, J. C., & van Meeuwen, L. W. (2024). AI chatbots in programming education: Students' use in a scientific computing course and consequences for learning. Computers and Education: Artificial Intelligence, 7, 100290.

Neumann, M., Rauschenberger, M., & Schön, E. M. (2023). "We need to talk about ChatGPT": The future of AI and higher education. In 2023 IEEE/ACM 5th International Workshop on Software Engineering Education for the Next Generation (SEENG)(pp. 29-32). IEEE. https://doi. org/10.1109/SEENG59157

Qadir, J. (2023, May). 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.

Vierhauser, M., Groher, I., Antensteiner, T., & Sauerwein, C. (2024, July). Towards integrating emerging ai applications in se education. In *2024 36th International Conference on Software Engineering Education and Training (CSEE&T)* (pp. 1-5). IEEE.

Walter, Y. (2024). Embracing the future of Artificial Intelligence in the classroom: the relevance of AI literacy, prompt engineering, and critical thinking in modern education. International Journal of Educational Technology in Higher Education, 21(1), 15.

Yelamarthi, K., Dandu, R., Rao, M., Yanambaka, V. P., & Mahajan, S. (2024). Exploring the potential of generative AI in shaping engineering education: Opportunities and challenges. Journal of Engineering Education Transformations, 439-445.