The Deskilling Effects of AI on Software Engineering and Education

I. Introduction

The rapid emergence of artificial intelligence (AI), particularly generative AI tools such as ChatGPT and GitHub Copilot, has transformed software engineering. While these tools promise increased productivity, they also raise concerns about deskilling—the erosion of human expertise as tasks become increasingly automated. Unlike previous technological shifts—such as the move from Assembly to C, or C to Java, which abstracted away tedious syntax while preserving logic—generative AI abstracts the logic generation itself. This shift fundamentally alters the cognitive load required to build software. This paper examines the deskilling effects of AI on software engineering and discusses implications for education and future learning.

II. Deskilling and AI in Software Engineering

Deskilling occurs when reliance on automation reduces opportunities to develop foundational skills. In software engineering, AI-assisted code generation can limit deep engagement with problem-solving, debugging, and system comprehension. Researchers describe this phenomenon as the creation of "capacity-hostile environments," in which technological convenience undermines long-term human skill development [1].

This phenomenon creates a specific instance of the Paradox of Automation [6] which I term the **"Code Reviewer's Paradox."** Historically, code review was a mentorship channel where senior engineers transferred wisdom to juniors. Today, as tools like GitHub Copilot offer automated code review and pull request summaries, we risk entering a "double-blind" loop where AI writes the code and AI reviews the code. While AI can catch syntax errors or style violations, it lacks the context to understand business logic or architectural intent. If a developer relies on an AI agent to "validate" their code, they may develop a false sense of security if silence from the AI is mistaken for correctness. This hollows out the critical skill of **auditability**—the human ability to look at a block of code and mentally trace its execution path. If we outsource both the writing and the reviewing, we remove the "human in the loop" entirely, turning software engineering into a "black box" operation where no human fully understands the system's behavior.

III. Productivity and Skill Formation

Empirical studies suggest that while AI tools can accelerate task completion, they may reduce conceptual understanding when overused. Developers relying heavily on AI-

generated code often demonstrate weaker retention of new programming concepts compared to those who engage directly with problem-solving [2].

This creates a cycle of **"cognitive offloading" [7]**. While offloading repetitive tasks (like boilerplate code) is beneficial, offloading core logic prevents the brain from forming the deep neural pathways associated with problem-solving. Even experienced developers must spend time validating and correcting AI output, underscoring the continued importance of human expertise [3]. If the foundational struggle of learning is removed, the expertise required to validate complex AI outputs may never form. Cognitive science suggests that deep learning occurs during **"productive struggle" [8]**—the frustration and iterative failure inherent in debugging and problem-solving. It is in these moments of friction that developers build robust mental models of system behavior, memory management, and edge cases. When AI removes this friction by providing instant solutions, it short-circuits the learning process, shifting the developer from a "builder" who understands the bricks and mortar to a "passenger" who enjoys the ride but cannot repair the engine.

IV. Labor Market Implications and Systemic Fragility

AI adoption has coincided with a measurable decline in entry-level software engineering roles, which traditionally serve as critical training grounds for novice developers [4]. This contraction is not merely technological but strategic. Major technology firms are increasingly leveraging AI to reduce reliance on human capital, prioritizing immediate cost efficiencies over long-term workforce sustainability.

By automating the entry-level tasks that once served as "apprenticeships"—such as writing documentation, unit tests, and boilerplate code—companies are effectively removing the bottom rungs of the career ladder. While this reduces operational costs in the short term, it creates a **"pipeline debt"** for the industry. If the junior roles where engineers learn to "break things safely" are eliminated, the industry faces a looming shortage of senior architects capable of managing complex, AI-integrated systems.

This "hollowed-out" workforce directly contributes to the rise of **fragile systems**. As developers increasingly rely on AI to generate entire modules of code they do not fully understand—a practice sometimes colloquially termed **"vibe coding"**—software becomes a "black box". When these systems inevitably fail, the lack of deep institutional knowledge makes remediation difficult or impossible. This aligns with recent findings on the **"trust calibration paradox" [9]**, where operators over-trust automated systems they lack the expertise to audit, leading to brittle infrastructure that is cheap to build but expensive to maintain.

V. Implications for Education

Educational institutions face growing challenges in aligning curricula with AI-augmented development practices. Overreliance on AI tools may weaken foundational learning if

students bypass critical reasoning and debugging processes. To combat this, Computer Science programs may need to shift assessment methods. To prove competency, students might need to face more oral exams and whiteboard coding without digital assistance, ensuring they possess the internalized knowledge to judge AI outputs effectively. Scholars argue that responsible AI integration should emphasize higher-order skills such as system design, ethical reasoning, and human-AI collaboration [5].

VI. Conclusion

AI is reshaping software engineering rather than eliminating it. While AI tools offer substantial benefits, unchecked reliance can contribute to deskilling. The future belongs not to the AI-dependent engineer, but to the AI-augmented engineer who maintains a discipline of manual practice. To mitigate these risks, educators must intentionally integrate AI into curricula in ways that preserve deep learning, critical thinking, and long-term skill development.

References

[1] V. Dignum, "Capacity-hostile environments and the ethics of artificial intelligence," AI & Society, 2025.

Available: https://link.springer.com/article/10.1007/s00146-025-02686-z

[2] E. Mollick and L. Mollick, "Using AI to enhance—not replace—learning," arXiv, 2024. Available: https://arxiv.org/abs/2601.08857

[3] M. Harris, "AI coding assistants may slow experienced developers," Reuters, 2024. Available: https://www.reuters.com/business/ai-slows-down-some-experienced-software-developers-study-finds-2025-07-10/

[4] Tom's Hardware Editorial Staff, "AI is eating entry-level coding jobs, Stanford study finds," Tom's Hardware, 2024.

Available: https://www.tomshardware.com/tech-industry/artificial-intelligence/ai-is-eating-entry-level-coding-and-customer-service-roles

[5] S. Noy and W. Zhang, "Experimental evidence on the productivity effects of generative artificial intelligence," Science, 2023.

Available: https://doi.org/10.1126/science.adh2586

[6] L. Bainbridge, "Ironies of automation," Automatica, vol. 19, no. 6, pp. 775–779, 1983.

[7] E. F. Risko and S. J. Gilbert, "Cognitive offloading," Trends in Cognitive Sciences, vol. 20, no. 9, pp. 676–688, 2016.

[8] E. L. Bjork and R. A. Bjork, “Making things hard on yourself, but in a good way: Creating desirable difficulties to enhance learning,” Psychology and the Real World, vol. 2, pp. 59–68, 2011.

[9] H. Zhang et al., “The Psychology of Learning from Machines: Anthropomorphic AI and the Paradox of Automation in Education,” arXiv preprint arXiv:2601.06172, 2026.

Available: https://arxiv.org/abs/2601.06172