Lessons Learned 6, 1 (2026)
Submitted: 10.11.2025
Accepted: 11.02.2026

DOI: https://doi.org/10.25369/ll.v6i1.109
ISSN: 2749-1307.

Zwischen Nutzen und Nebenwirkungen des ChatGPT-Einsatzes beim Programmieren – eine Pilotstudie im Lehr-Lern-Labor-Technik

I. Gideon, N. Link

Professur für Technik und Technische Bildung, Institut für Physik und Technische Bildung, Fakultät für Natur- und Sozialwissenschaften, Pädagogische Hochschule Karlsruhe

Abstract

Generative KI-Chatbots wie ChatGPT zeigen neue Möglichkeiten zur Unterstützung von Lernprozessen in der Programmierausbildung, deren Wirksamkeit jedoch von der didaktischen Einbettung abhängt. In einer Pilotstudie entwickelten Studierende ein Arduino-Projekt mithilfe von ChatGPT. Analysiert wurden sowohl die erstellten Programme der Studierenden als auch ihre Logbücher zur Promptnutzung. Die Logbücher wurden anhand eines literaturgestützten Kategoriensystems effektiver Prompts kodiert. Die Ergebnisse zeigen, dass die Prompts meist unpräzise, kaum kontextualisiert und selten iterativ weiterentwickelt wurden. Studierende übernahmen den generierten Programmcode häufig, ohne dessen Funktionslogik zu verstehen, was sich in Schwierigkeiten beim Erkennen und Beheben von Codefehlern zeigte. Die Befunde deuten darauf hin, dass eine rein ChatGPT-basierte Programmierung den Wissenserwerb von Novizinnen und Novizen kaum fördert oder ihn erschweren kann.

Generative AI chatbots such as ChatGPT offer new possibilities for supporting learning processes in programming education, but their effectiveness depends on their didactic integration. In a pilot study, students developed an Arduino project with the help of ChatGPT. Both the programs created by the students and their logbooks on prompt usage were analyzed. The logbooks were coded using a literature-based category system of effective prompts. The results show that the prompts were mostly imprecise, hardly contextualized, and rarely developed iteratively. Students often adopted the generated program code without understanding its functional logic, which was reflected in difficulties in recognizing and correcting code errors. The findings suggest that programming based solely on ChatGPT does little to promote knowledge acquisition among novices and may even hinder it.

*Corresponding author: igor.gideon@ph-karlsruhe.de

1. Ausgangslage

Das Verständnis der Funktionsweise von Software und das Schreiben von Programmen sind häufige Anforderungen von technischen Berufen; die Komplexität des Programmierlernens führt oft zu Bildungsmisserfolgen und Frustration der Lernenden [1]. Generative KI-Chatbots wie ChatGPT finden in der Programmierausbildung zunehmend Anwendung [2, 3], da sie neuartige und vielversprechende Möglichkeiten zur Unterstützung von solchen Lernprozessen eröffnen [4, 5]. Neben der reinen Codegenerierung können sie Erklärungen liefern, Syntaxfehler identifizieren oder alternative Lösungsansätze anbieten [6]. Ihre Nutzung wird mittlerweile auch in der fachdidaktischen Forschung zunehmend thematisiert [7, 8]. Insgesamt entwickelt sich die Integration von OpenAI in den Programmierunterricht zu einem allgemeinen Trend, bei dem Chatbots, intelligente Tutor- und automatisierte Bewertungssysteme personalisierte Unterstützung und Feedback ermöglichen [1, 9].

Auch Co-Programmierungswerkzeuge gewinnen an Bedeutung, die generative KI direkt in Entwicklungsumgebungen integrieren (z. B. Cursor/X, Visual Studio Code mit GitHub Copilot oder Entwicklungsumgebungen mit Agenten wie Antigravity oder Gemini) und damit eine hohe Aufgaben- und Prozessintegration im Programmierworkflow ermöglichen [23]. In wissenschaftlichen Studien wird diese eng verzahnte Form der Mensch-KI-Zusammenarbeit als „Cyborg"-Ansatz beschrieben, bei dem der Arbeitsfluss kontinuierlich mit KI-Interaktionen verflochten ist (im Gegensatz zu eher arbeitsteiligen „Centaurs") [23].

Typische Stärken generativer KI-Chatbots liegen in der natürlichen Sprachverarbeitung und der Möglichkeit personalisierter Rückmeldungen [10, 11, 12]. Zudem werden Modelle alternativer Anbieter (z. B. Anthropic Claude Opus medium) in öffentlichen Performanzvergleich (z. B. SWE-bench) häufig als sehr leistungsfähig ausgewiesen.

Dennoch weist ihre Nutzung auch Einschränkungen auf, etwa in Form fehlerhafter Codes, oberflächlicher Logik oder ethischer Art [10, 12, 13]. Hohe Nutzenerwartungen an die KI bei gleichzeitig wahrgenommenen Risiken bestätigen diese Ambivalenz [14].

Im Bildungskontext zeigt sich ein ähnliches Spannungsfeld: Während manche Studierende von individuell empfundenen Effizienzsteigerungen und Lernfortschritten berichten, beobachten Lehrende Qualitäts- und Kompetenzverluste, die durch Nutzung von OpenAI bei den Lernenden entstehen [2]. Häufig zeigt sich, dass durch die Nutzung von OpenAI ein eher rezeptiver Nutzungsstil begünstigt wird, indem Studierende zur Übernahme des generierten Codes durch Copy-Paste tendieren, ohne dessen Funktionsweise zu durchdringen [3, 8]. Besonders kritisch erscheint der Befund, der auf einen deutlichen Rückgang des kollaborativen Lernens (Interaktion und Diskussion) durch Nutzung von ChatGPT hinweist [2].

Obwohl in der Literatur etablierte Leitlinien für lernförderliches Prompten vorliegen, werden diese in der Praxis bislang nur selten konsequent umgesetzt. Dazu zählen etwa einfache Gedankenketten-Aufforderungen (z. B. vom Allgemeinen zum Konkreten), die die Korrektheit und Qualität der KI-Antworten erhöhen [6], sowie das häufig zitierte *Five-S-Rahmenmodell* (Kontextualisierung, Präzision, Strukturierung, Metaorganisation, Iteration) als praktikable Heuristik zur Verbesserung der Outputqualität [8]. Studien berichten, dass Lernende ohne didaktische Rahmung die grundlegenden Aspekte des erfolgreichen Promptens übersehen, sodass die eigentliche Lernprogression zur Passivierung tendiert [2]. Dies führt folgerichtig zu Defiziten in der Fähigkeit, den Programmcode zu verstehen, kritisch zu analysieren oder zu modifizieren [2]. Auf diese Weise können die Potenziale generativer KI-Chatbots weniger als Lernhilfe und mehr als verkürzter Lösungsweg genutzt werden [2]. Forschende warnen in diesem Zusammenhang vor einem „blinden Vertrauen", da die einfache Möglichkeit, sich von ChatGPT Lösungen bereitstellen zu lassen, die Entwicklung des eigenen kritischen Denkens und eigenständiger Problemlösefähigkeit beeinträchtigen kann [12]. Stattdessen sollte Wissen durch Dialog erworben werden, in diesem Kontext also durch Feedback-Interaktionen mit ChatGPT [15]. Die bisherigen Forschungsergebnisse deuten insgesamt eher darauf hin, dass ChatGPT beim Programmieren ohne pädagogische Maßnahmen nur eingeschränkt lernförderlich ist [16, 17].

Eine mögliche Ursache hierfür könnte in einem unzureichenden Verständnis der Funktionsweise von KI-Systemen liegen – insbesondere darüber, welche Eingaben für Lernprozesse erforderlich sind, nicht nur zur Erzeugung von Outputs. Fehlendes Wissen über die konkrete Formulierung einzelner Prompts sowie über die metakognitive Fähigkeit, diese strategisch einzusetzen, erschwert also einen lernwirksamen Umgang mit KI. Daher wird empfohlen, den Umgang mit Prompts sowie die Reflexion über die Grenzen von KI explizit als Lerninhalt zu gestalten, um der Entstehung möglicher Kompetenzdefizite vorzubeugen [4, 12, 14].

2. Forschungsfragen

Da bislang keine Untersuchungen zur Lernwirksamkeit generativer KI-Chatbots in der Lehramtsausbildung im Kontext der technischen Bildung vorliegen, ergibt sich ein Forschungsdesiderat. Zu klären ist, wie KI-Chatbots in Lernprozesse von Lehramtsstudierenden so integriert werden können, dass sie Kompetenzentwicklung fördern.

Um diese Frage zu bearbeiten, erscheint es zunächst erforderlich zu prüfen, ob Lehramtsstudierende, die mithilfe von ChatGPT intuitiv Programmieren lernen, tatsächlich nur begrenzte Programmierfähigkeiten erwerben. Darüber hinaus ist zu untersuchen, wie Studierende ihre Prompts zur Generierung von Programmcode formulieren und in welchem Zusammenhang die Qualität dieser Prompts mit ihren Programmierfähigkeiten sowie mit der Qualität des resultierenden Codes steht. Auf dieser Grundlage werden folgende zwei Forschungsfragen formuliert:

1. *Wie formulieren Studierende Prompts in ChatGPT, um Programmcode zu erhalten?*
2. *Wie hängt die Formulierung der Prompts mit der Programmierfähigkeit und der Qualität der Codefunktionalität zusammen?*

3. Pilotstudie

Aufgabenstellung

In einer Lehrveranstaltung im Fach Technik (Sekundarstufe I) wurde im Februar 2025 eine Pilotstudie durchgeführt, in der Lehramtsstudierende (N = 13), überwiegend ohne bzw. mit geringen Programmierkenntnissen, eigene Projekte entwickelten. Die Aufgabe bestand darin, mit einem Arduino Uno R4 in der Online-Anwendung TinkerCAD ein technisches System zu konzipieren und umzusetzen. Dieses sollte mindestens zwei Sensoren und zwei Aktoren enthalten sowie entsprechende Mess-, Regel- und Steuerfunktionen realisieren.

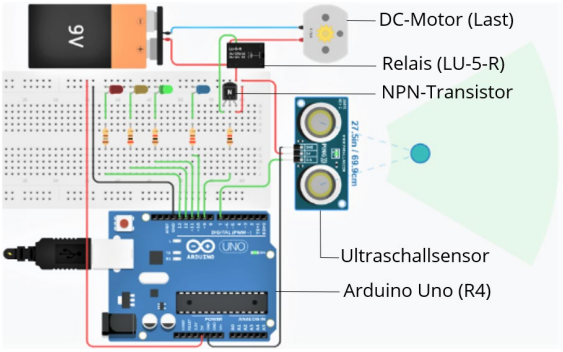

Abb. 1: TinkerCAD-Simulation eines Arduino-Uno-Projekts zur Distanzmessung und Motorsteuerung

Ein Beispielmodell (Abb. 1) umfasst die Füllstandsmessung mittels Ultraschallsensor, eine LED-Anzeige in Ampellogik sowie die Ansteuerung eines DC-Motors über einen Transistorschalter in Abhängigkeit vom gemessenen Füllstand, wobei der Motor über ein vom Transistor angesteuertes Relais geschaltet wird, sodass der Arduino nur den Steuerstrom liefert und der Motorstrom separat über die 9 V-Versorgung fließt. Die LEDs visualisieren dabei drei Zustände: Grün signalisiert einen unkritischen Bereich (kleiner Abstand, hoher Füllstand), Gelb einen Vorwarnbereich (zunehmender Abstand, sinkender Füllstand) und Rot einen kritischen Bereich (großer Abstand, niedriger Füllstand), in dem der Motor – beispielweise als Antrieb einer Wasserpumpe – aktiviert wird. Ergänzend ist eine blaue Status-LED vorgesehen, die den aktiven Pumpenbetrieb unabhängig von der Ampelanzeige kenntlich macht, etwa an einer räumlich getrennten Stelle als Betriebsanzeige. Der Abstand, der rechts vom Ultraschallsensor im Modell durch einen blauen Marker visualisiert ist, wird in der Lernumgebung manuell variiert, um unterschiedliche Füllstände zu simulieren. Zur Erstellung eines entsprechenden Programmiercodes war den Teilnehmenden die Nutzung von ChatGPT und weiteren Online-Ressourcen ausdrücklich erlaubt. Studierende nutzten

überwiegend die kostenfreie ChatGPT-Version. Hierbei standen ihnen GPT-4o (mit Nutzungslimits) sowie o3-mini zur Verfügung; bei ausgeschöpften Limits konnten GPT-4o mini genutzt werden.

Während der Aufgabenbearbeitung führten die Studierenden ein Logbuch, in dem sie an ChatGPT gerichteten Prompts dokumentierten.

4. Methode

Die Pilotstudie folgte einem Mixed-Methods-Ansatz. Neben der Analyse des erstellten Codes und der Fähigkeit, die Logik dieser Codes zu verstehen, wurden die von Studierenden dokumentierten Prompts ausgewertet. Ergänzend wurden Fragebögen eingesetzt, um Selbstauskunft zu Programmier- und ChatGPT-Vorkenntnissen zu erfassen.

Die Studie zielt dabei primär auf die Erfassung von Performanz-Indikatoren während der Aufgabenbearbeitung (produkt- und verstehensorientierte Anteile der Programmierfähigkeit) und nicht auf die Quantifizierung von Lernzuwächsen im Sinne eines Vorher-Nachher-Vergleichs. Aussagen zum „Kompetenzerwerb" beziehen sich daher auf Hinweise aus dem beobachtbaren Verstehen, Erklären und Modifizieren im Aufgabenprozess.

Analyse der Selbstauskunft über Vorkenntnisse

Zur Einschätzung der Vorkenntnisse der Studierenden wurden ihre Selbstauskünfte zum Umgang mit OpenAI sowie zu Programmierkenntnissen erhoben. Zwei Personen wurden von der weiteren Analyse ausgeschlossen, da sie in der Selbstauskunft angaben, über fortgeschrittene Programmierkenntnisse zu verfügen und die Studie sich aber an Novizinnen und Novizen richtet.

Der eingesetzte Selbsteinschätzungstest bildet die Dimensionen (1) „Grundlagen der Programmierung" und (2) „Umgang mit OpenAI bzw. ChatGPT" ab. Die erste Dimension umfasst Aspekte wie das Erstellen einfacher block- oder textbasierter Programme (z. B. „Ich kann einfache Programme textbasiert erstellen."), das Verständnis grundlegender Strukturen wie Schleifen, Bedingungen, Variablen sowie Fehlersuche und Testläufe. Die zweite Dimension erfasst die Selbsteinschätzung zum Verständnis und zur Nutzung von ChatGPT. Sie umfasst Wissen über Funktionsweise und Grenzen, die Fähigkeit zur Prompt-Formulierung, die Einschätzung der Antwortqualität sowie eine reflektierte Anwendung in unterschiedlichen Kontexten (z. B. „Ich nutze systematische Methoden, z. B. Schritt-für-Schritt-Erklärung, um bessere Antworten zu bekommen.").

Die Teilnehmenden bewerteten ihre Fähigkeiten entlang einer fünfstufigen Likert-Skala (1 = unsicher bis 5 = sicher). Zur weiteren Analyse wurden die fünf Antwortstufen zunächst auf eine lineare Skala von 0 bis 1 transformiert (unsicher = 0; eher unsicher = .25; neutral = .5; eher sicher = .75; sicher = 1). Auf dieser Basis wurden die Mittelwerte beider Dimensionen ermittelt.

Analyse der Prompt-Formulierungen

In Anlehnung an das etablierte Five-S-Rahmenmodell [8] wurde für die Analyse der Prompts ein Kategoriensystem herangezogen. Es umfasst insgesamt sieben Dimensionen, die zentrale Empfehlungen zum effektiven Prompting mit KI-Systemen wie ChatGPT systematisieren:

- *Kontext:* Strategien zur Herstellung eines inhaltlichen oder sozialen Bezugsrahmens. Dazu zählen etwa das Einbetten von Situationen sowie die Formulierung kontextualisierter oder orientierender Prompts [18]. Ergänzend können Rollen- und Persona-Zuweisungen [5, 17] sowie Integration von Beispielen, externen Informationen die Anschlussfähigkeit des Chatbots erhöhen [19].
- *Präzision:* Techniken zur sprachlichen und thematischen Schärfung. Damit gemeint sind explizite Ausdrucksweisen, sprachliche Vereinfachungen sowie die Konkretisierung der Fragestellungen [8].
- *Strukturierung:* Maßnahmen, die eine klare und logische Gliederung der KI-Ausgaben fördern. Hierzu zählen schrittweise Anleitungen, Strategien zum Umgang mit Mehrdeutigkeiten sowie Techniken wie „Zero-Shot Chain-of-Thought" (z. B. „Denke Schritt für Schritt nach"), die explizite Denkstrukturen anregen [6, 22].
- *Metaorganisation:* Übergeordnete Steuerungsmechanismen im Prompting-Prozess. Beispiele sind die Differenzierung von

Aufgabenarten, das Festlegen von Lernstilen, das Anlegen eigener Formatlogik oder die Aufforderung zu spezifischen Visualisierungen [6, 19].

- *Iteration:* Formen des sequenziellen Promptens, bei denen Folgeeingaben gezielt an vorherige Ausgaben anschließen. Hierzu zählen „Prompt Chaining", „Dialogisches Prompting" und „Flipped Interaction", bei denen Antworten schrittweise weiterentwickelt und in eine anschlussfähige Abfolge überführt werden. Auch das Einfordern von Rückmeldungen fällt in diesen Bereich [5].
- *Normativ-ethische Aspekte:* Querschnittsthemen wie ethische Fragestellungen oder das Spannungsfeld zwischen Nutzerintention und KI-generierter Kreativität [5].
- *Ineffektives Prompting:* Typische Fehlformen des Promptings, etwa unklare Formulierungen, unrealistische Erwartungen an die KI oder eine übermäßige Detailtiefe [17].

Insgesamt umfasst das Kategoriensystem 28 Einzelkategorien, deren Verteilung sich wie folgt über die sieben Dimensionen erstreckt: Kontext (7), Präzision (5), Strukturierung (4), Metaorganisation (4), Iteration (3), Ineffektives Prompting (3) und Normativ-ethische Aspekte (2) (siehe Anhang 1).

Die Bewertung des Promptens erfolgte kriterienorientiert anhand einer dichotomen Skala, indem für jede Kategorie ein Code vergeben wurde (1 = trifft zu, 2 = trifft nicht zu). Eine exklusive Zuordnung zu nur einer Kategorie erfolgte nicht. Anschließend wurden die Häufigkeiten der Codes je Kategorie berechnet.

Interrater-Reliabilität

Zur Überprüfung der Zuverlässigkeit der vorgenommenen Codierungen wurde eine Interrater-Reliabilitätsanalyse durchgeführt. Dafür wurden 50 % der Daten von zwei unabhängigen Ratern anhand des Kategorienschemas doppelt codiert. Zur Bestimmung der Gesamtkonsistenz wurde ein globaler Kappa-Wert (k) nach Cohen [20] über alle doppelt codierten Einheiten berechnet. Der resultierende Wert k = 0.76 weist nach Landis und Koch [21] auf eine substanzielle Übereinstimmung zwischen den Ratern hin.

Analyse der Programmierfähigkeit

Unter Programmierfähigkeit wird in dieser Studie die Kompetenz verstanden, nicht nur funktionsfähigen Code zu erstellen bzw. durch ChatGPT generieren zu lassen und anzuwenden, sondern auch dessen Logik nachvollziehen und erläutern zu können [11]. Dieses Konstrukt wurde in zwei Teilaspekte untergliedert:

(1) *Produktorientiert* durch die in TinkerCAD erstellten Programme der Studierenden, die Aufschluss über syntaktische Korrektheit und funktionale Umsetzbarkeit geben.

(2) *Verstehensorientiert*, da die reine Codeerstellung nicht zwangsläufig die zugrunde liegende Programmierfähigkeit widerspiegelt [22]. Zu diesem Zweck wurden die Studierenden aufgefordert, die Funktionsweise ihres Codes sowie die Zusammenhänge einzelner Codezeilen und -abschnitte zu erläutern. Enthielten ihre Programmcodes Fehler, sollten diese identifiziert und korrigiert werden.

Damit sollte eine mögliche Diskrepanz zwischen „funktionierendem Ergebnis" und „konzeptuellem Durchdringen" abbildbar werden. Einschränkungen im Erklären und Modifizieren wurden entsprechend als Hinweise auf begrenzte verstehensorientierte Kompetenz(entwicklung) im Aufgabenprozess interpretiert.

5. Ergebnisse

Selbstauskunft über Vorkenntnisse

Die Selbsteinschätzungen der Teilnehmenden zeigen, dass sie im Mittel über mittlere Vorerfahrungen im Umgang mit OpenAI-Anwendungen verfügten (M = .77; SD = .19)[1], während sie ihre Programmierkenntnisse als gering einschätzten (M = .34; SD = .30).

Forschungsfrage 1

Die Analyse der von Studierenden erstellten Prompts zeigt, dass in der Literatur empfohlene Prompting-Prinzipien nur selten umgesetzt wurden. Dazu zählen insbesondere die *Five-S-Strategien* nach Tassoti [8] sowie weitere Best Practices des Prompt Engineerings [5, 6, 17, 18]. Über alle Codierungen hinweg erfüllten lediglich etwa

- 11 % der beobachteten Kategorien die Kriterien der jeweiligen Prinzipien, während in

[1] M = Mittelwert; SD = Standardabweichung

- rund 89 % der Fälle keine Anwendung davon identifizierbar war. Stattdessen überwog meist eine einmalige Prompteingabe der Studierenden mit einer anschließenden Übernahme der generierten Programmcodes.

Da die Teilnehmenden in diese Techniken nicht eingeführt wurden, erfolgte das Prompten vermutlich weitgehend intuitiv.

Forschungsfrage 2

Die Ergebnisse der Pilotstudie zeigen zudem, dass der von den Studierenden mithilfe von ChatGPT generierte und übernommene Code in der Regel funktionsfähig war und die Modelle korrekt ansteuerte. Deutliche Schwierigkeiten traten jedoch dann auf, wenn Anpassungen erforderlich waren und die Lernenden den Code selbstständig modifizieren sollten.

Insgesamt fiel es den Studierenden schwer, den Überblick über den Code zu behalten und die logischen Zusammenhänge zwischen einzelnen Zeilen oder Abschnitten zu erläutern.

Bezogen auf die in Kapitel 4 festgelegte Operationalisierung der Programmierfähigkeit (produkt- und verstehensorientiert) zeigt sich damit eine Diskrepanz: Während die produktorientierten Ergebnisse gelingen, bleibt der verstehensorientierte Anteil in vielen Fällen eingeschränkt.

6. Diskussion

Die in dieser Pilotstudie beobachteten Ergebnisse legen nahe, dass die Potenziale von ChatGPT häufig auf Kosten einer aktiven Auseinandersetzung mit Programmierinhalten genutzt wurden. Dies könnte zu einer Passivierung des Lernprozesses geführt und die Fähigkeit zur Fehleranalyse und -korrektur eingeschränkt haben: Viele Teilnehmende waren nicht in der Lage, die Mängel im generierten Code zu identifizieren und zu beheben.

Damit bestätigt sich, dass ChatGPT zwar als unterstützendes Werkzeug im Programmierunterricht dienen kann, für Novizinnen und Novizen jedoch kein Ersatz für bisherige fundierte didaktische Konzepte ist [2]. Entsprechend ist der Befund nicht als „Defizit“ der Studierenden zu interpretieren, sondern als eine mögliche Folge fehlender Instruktion. Für die Einordnung des (begrenzten) Kompetenzerwerbs ist jedoch relevant, dass der beobachtete Nutzungsstil von ChatGPT die Anzahl an Lerngelegenheiten reduzierte, in denen Lernende den Code schrittweise erklären lassen, begründet variieren, Fehler diagnostizieren oder iterativ verbessern.

Der in dieser Arbeit beobachtete Einsatz birgt die Gefahr, Abhängigkeiten von externem Wissen zu verstärken, statt Kompetenzen aufzubauen. Daraus ergibt sich die Notwendigkeit, das Prompting als eigenständige Kompetenz systematisch zu fördern.

Die Literatur stützt diese Annahme allerdings nicht konsistent: Sagar [17] berichtet von Fällen, in denen Lernende ohne strukturiertes Prompting-Framework bessere Ergebnisse erzielten als Lernende, die mit einem Framework lernten. Mögliche Ursachen hierfür könnten aber in der Eignung des eingesetzten Frameworks oder in unterschiedlichen Lernstrategien der Teilnehmenden liegen. Auch Studien im Bereich der Physikdidaktik zeigen, dass die Qualität von ChatGPT-Ausgaben stark vom eingesetzten Prompting abhängt und durch gezielte Techniken erheblich verbessert werden kann [6]. Weitere Forschung sollte daher klären, wie Frameworks gestaltet sein müssen, um unterschiedliche Lernstrategien produktiv zu unterstützen.

Für die Lehramtsausbildung im Fach Technik folgt daraus, dass nicht nur die Möglichkeiten, sondern auch die didaktischen Grenzen generativer KI reflektiert berücksichtigt werden müssen. Der Aufbau von Prompt-Kompetenz sollte als Lernziel verankert werden. Zudem bietet sich an, eine Progression zu etablieren, die von der Basishandhabung über das Analysieren fremder Lösungen hin zur selbstständigen Entwicklung und Optimierung der Prompts führt.

Ohne gezielte Anleitung besteht die Gefahr, dass KI-Assistenz die Eigenaktivität und Kompetenzentwicklung hemmt. Eine Einbettung in didaktisch gestaltete Lernumgebungen kann dem entgegenwirken, indem sie Problemlösekompetenz, strategisches Vorgehen und Wissenserwerb fördert. Künftige Forschung sollte daher untersuchen, wie KI-gestützte Programmierassistenz so in Lernprozesse integriert werden kann, dass diese förderlichen Effekte tatsächlich realisiert werden.

7. Limitationen

Die Aussagekraft dieser Pilotstudie ist durch mehrere Einschränkungen begrenzt.

Erstens ist der Stichprobenumfang klein (N = 13) und auf Lehramtsstudierende in einem spezifischen Kontext (Arduino Uno in TinkerCAD) beschränkt. Dadurch ist die Übertragbarkeit der Befunde auf andere Zielgruppen, Lernumgebungen oder reale Hardware-Setups zu hinterfragen.

Zweitens fehlte eine Vergleichsbedingung mit etwa einer Einführung in effektive Prompting-Strategien als variierender Faktor. Dadurch lässt sich nicht eindeutig bestimmen, ob die beobachteten Schwierigkeiten auf die Nutzung der KI selbst oder auf die fehlende didaktische Rahmung beim Einsatz solcher Werkzeuge zurückzuführen sind.

Drittens beruhte die Analyse der Promptqualität auf einem dichotomen Codierschema (1 = trifft zu, 2 = trifft nicht zu). Diese Binarität reduziert graduelle Unterschiede und kann feine qualitative Abstufungen der Promptqualität nivellieren.

Viertens wurde die Mensch-KI-Zusammenarbeit in dieser Studie primär als „KI als Team-Kollege" (im Sinne eines Gegenübers) realisiert. Andere Kollaborationsmodi, insbesondere stark integrierte Co-Working-Ansätze im Sinne eines „*Cyborg*"-Modus [23] bei dem Mensch und KI verzahnt und iterativ arbeiten, wurden nicht umgesetzt. Daher ist angesichts des in dieser Arbeit skizziertes Five-S-Rahmenmodells [8] zu berücksichtigen, dass dieses Rahmenmodell primär für klassische Chatbot-Interaktionen entwickelt wurde. Bei anderen Modellen, z. B. mit expliziten Reasoning-Mechanismen können identische Prompts andere Antwortstrategien auslösen; die in Kapitel 6 diskutierte Inkonsistenz in der Literatur ist deshalb vor diesem Hintergrund zu interpretieren.

Die Befunde dieser Arbeit sind als Tendenzen zu verstehen und liefern erste Hinweise auf Prompting-Praktiken sowie Herausforderungen im KI-gestützten Programmierlernen von Lehramtsstudierenden im Fach Technik. Ihre Bestätigung und Generalisierung erfordern weiterführende Studien mit größeren und heterogeneren Stichproben sowie kontrollierten Designs, um die beobachteten Zusammenhänge systematisch zu prüfen und differenzierter zu erfassen. Dabei könnte auch die Art der Mensch-KI-Kollaboration (z. B. *Centaur* vs. *Cyborg*) als experimenteller Faktor berücksichtigt werden.

Autorenbeiträge

Igor Gideon:
Konzeptualisierung, Methodik, formale Analyse, Untersuchung (Datenerhebung), Datenkuratierung, Verfassen des Originalentwurfs, Visualisierung.

Nico Link:
Konzeptualisierung, Methodik, Begutachtung und Überarbeitung (Review & Editing), Supervision.

Für die sprachliche Überarbeitung wurde das Tool DeepL (Version 25.11.4) eingesetzt; die inhaltliche Verantwortung liegt bei den Autoren.

Literatur

[1] Skalka, J., Drlik, M., Benko, L., Kapusta, J., Rodriguez del Pino, J. C., Smyrnova-Trybulska, E., ... Turcinek, P. (2021). Conceptual framework for programming skills development based on microlearning and automated source code evaluation in virtual learning environment. Sustainability, 13(6), 3293. https://doi.org/10.3390/su13063293

[2] 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. https://doi.org/10.1016/j.caeai.2024.100290

[3] Sun, D., Boudouaia, A., Zhu, C., & Li, Y. (2024). Would ChatGPT-facilitated programming mode impact college students' programming behaviors, performances, and perceptions? An empirical study. International Journal of Educational Technology in Higher Education, 21(1), 14. https://doi.org/10.1186/s41239-024-00446-5

[4] Leifheit, L., Loefflad, D., Belschner, S., Beuttler, B., Winkelmann, J., Meurers, W. D., & Holz, H. (2024). KI im Unterricht: Entwicklung von Lehrveranstaltungen für Lehramtsstudierende der Sprach- und MINT-Fächer. Ludwigsburger Beiträge zur Medienpädagogik, 24, 1–19. https://doi.org/10.21240/lbzm/24/08

[5] White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., & Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with ChatGPT. arXiv. https://doi.org/10.48550/arXiv.2302.11382

[6] Polverini, G., & Gregorcic, B. (2024). How understanding large language models can inform the use of ChatGPT in physics education. European Journal of

Physics, 45(2), 025701. https://doi.org/10.1088/1361-6404/ad1b3a

[7] Greco, C. M., & Tagarelli, A. (2024). Bringing order into the realm of Transformer-based language models for artificial intelligence and law. Artificial Intelligence and Law, 32(4), 863–1010. https://doi.org/10.1007/s10506-023-09372-x

[8] Tassoti, S. (2024). Assessment of students' use of generative artificial intelligence: Prompting strategies and prompt engineering in chemistry education. Journal of Chemical Education, 101(6), 2475–2482. https://doi.org/10.1021/acs.jchemed.3c01069

[9] Malik, S. I., Ashfque, M. W., Tawafak, R. M., Al-Farsi, G., Usmani, N. A., & Khudayer, B. H. (2022). A chatbot to facilitate student learning in a Programming 1 course: A gendered analysis. International Journal of Virtual and Personal Learning Environments, 12(1), 1–20. https://doi.org/10.4018/IJVPLE.310007

[10] Yılmaz, R., & Yılmaz, F. G. K. (2023). Augmented intelligence in programming learning: Examining student views on the use of ChatGPT for programming learning. Computers in Human Behavior: Artificial Humans, 1(2), 100005. https://doi.org/10.1016/j.chbah.2023.100005

[11] Wang, T., Díaz, D. V., Brown, C., & Chen, Y. (2023). Exploring the role of AI assistants in computer science education: Methods, implications, and instructor perspectives. In 2023 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC) (pp. 92–102). IEEE. https://doi.org/10.1109/VL-HCC57772.2023.00018

[12] Rahman, M. M., & Watanobe, Y. (2023). ChatGPT for education and research: Opportunities, threats, and strategies. Applied Sciences, 13(9), 5783. https://doi.org/10.3390/app13095783

[13] Chen, E., Huang, R., Chen, H. S., Tseng, Y. H., & Li, L. Y. (2023). GPTutor: A ChatGPT-powered programming tool for code explanation. In International Conference on Artificial Intelligence in Education (pp. 321–327). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-36336-8_50

[14] Samek, W., & Schmid, U., et al. (2025). Nachvollziehbare KI: Erklären, für wen, was und wofür (Whitepaper der Plattform Lernende Systeme). Plattform Lernende Systeme / acatech. https://doi.org/10.48669/pls_2025-2

[15] Beyerer, J., Kirchner, E., André, E., Behnke, S., Bloss, G., Dzaack, J., Egloffstein, T., Griepentrog, H. W., Gustmann, M., Hafner, V. V., Koert, D., Nüchter, A., Seyler, J., Straube, S., Tchouchenkov, I., von Stryk, O., Wedler, A., Wolf-Ostermann, K., & Zimmermann, M. (2025). KI in der Robotik: Flexible und anpassbare Systeme durch interaktives Lernen (Whitepaper der Plattform Lernende Systeme). Plattform Lernende Systeme / acatech. https://doi.org/10.48669/pls_2025-1

[16] Stoyanova, D., Stoyanova-Petrova, S., & Mileva, N. (2025). Exploring students' and teachers' perceptions about using ChatGPT in programming education. International Journal of Engineering Pedagogy, 15(2).

[17] Sagar, P. (2024). Exploring the use of ChatGPT and the prompting framework as a self-learning aid for Arduino coding & circuit building for artists and designers (Doctoral dissertation, Massachusetts Institute of Technology). MIT Libraries. https://hdl.handle.net/1721.1/151432

[18] Ekin, S. (2023). Prompt engineering for ChatGPT: A quick guide to techniques, tips, and best practices. TechRxiv. https://doi.org/10.36227/techrxiv.22683919.v1

[19] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., … Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. In Advances in Neural Information Processing Systems (NeurIPS 35) (pp. 24824–24837). https://proceedings.neurips.cc/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html

[20] Cohen, J. (1960). A coefficient of agreement for nominal scales. Educational and Psychological Measurement, 20(1), 37–46. https://doi.org/10.1177/001316446002000104

[21] Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. Biometrics, 33(1), 159–174. https://doi.org/10.2307/2529310

[22] Röhner, G. (2016). Bildungsstandards Informatik für die Sekundarstufe II (Januar 2016). Gesellschaft für Informatik e. V. https://doi.org/10.18420/rec2016_057

[23] Dell'Acqua, F., McFowland III, E., Mollick, E. R., Lifshitz-Assaf, H., Kellogg, K., Rajendran, S., … & Lakhani, K. R. (2023). Navigating the jagged technological frontier: Field experimental evidence of the effects of AI on knowledge worker productivity and quality. Harvard Business School Technology & Operations Mgt. Unit Working Paper, (24-013).

Anhang

Anhang 1. Übersicht des Kategoriensystems „Prompting-Prinzipien“ mit eigener Zuordnung zu den sieben Dimensionen.

Nr.	Kategorie	Autoren	Subkategorie	Beschreibung
1	Kontext	Prompting-Strategien (Five S, Tassoti 2024)	Set the Scene (Kontext geben)	Wird ein konkreter Hintergrund oder eine spezifische Situation beschrieben?
2	Kontext	Prompt Engineering (Giray, 2023)	Kontextuelle Prompts	Enthält der Prompt zusätzliche Hintergrundinformationen, um Antwort auf spezifische Aspekte zu fokussieren?
3	Kontext	Prompt-Komponenten (Polverini & Gregorcic, 2024; Sagar, 2024)	ChatGPT eine Persona zuweisen („Assign ChatGPT a Persona“)	Wird ChatGPT eine explizite Rolle zugewiesen, um Tiefe und Kontext der Antworten zu beeinflussen?
4	Kontext	Prompt-Komponenten (Sagar, 2024)	Eigene Persona definieren („Define Your Persona“)	Wird explizit angegeben, welche Rolle oder welches Kompetenzniveau der Fragesteller hat?
5	Kontext	Prompt Engineering (Ekin, 2023) – Techniken	Illustration durch Beispiele	Werden Beispiele genutzt, um die Genauigkeit der Aufgabenstellung zu erhöhen?
6	Kontext	Prompt Engineering (Polverini & Gregorcic, 2024)	Few-Shot CoT	Werden Beispiele genutzt, um Lösung von Aufgaben zu illustrieren?
7	Kontext	Prompt Engineering (Ekin, 2023) – Best Practices	Integration externer Daten	Werden externe Quellen oder Daten explizit in den Prompt integriert?
8	Präzision	Prompt Pattern Catalog (White et al., 2023)	Motivation	Wird angegeben, was mit dem Prompt erreicht werden soll?
9	Präzision	Prompting-Strategien (Five S, Tassoti 2024)	Simplify your Language (Sprache vereinfachen)	Wird eine einfache, verständliche Sprache gefordert?
10	Präzision	Prompt-Komponenten (Sagar, 2024)	Thema konkretisieren („Elaborate Topic of Interest“)	Wird das Lernziel klar formuliert?
11	Präzision	Prompt Engineering (Ekin, 2023; Giray, 2023; Five S, Tassoti 2024)	Klare und spezifische Anweisungen	Werden Missverständnisse durch explizite, spezifische Anweisungen vermieden?
12	Präzision	Prompt Pattern Catalog (White et al., 2023)	Question Refinement	Fordert der Prompt explizit eine Verfeinerung oder Optimierung der gestellten Fragen?
13	Strukturierung	Prompting-Strategien (Polverini & Gregorcic, 2024; Tassoti 2024)	Structure the Output (Ausgabe strukturieren)	Wird eine strukturierte, schrittweise bzw. gegliederte Antwort verlangt?
14	Strukturierung	Prompt Pattern Catalog (White et al., 2023)	Output Automater	Verweist der Prompt auf spezifische Antwortmuster?
15	Strukturierung	Prompt Engineering (Ekin, 2023) – Techniken	Kontrolle der Ausführlichkeit	Wird gezielt nach einer bestimmten Detailtiefe oder Kürze verlangt?
16	Strukturierung	Prompt Engineering (Ekin, 2023) – Fortgeschrittene Strategien	Umgang mit Mehrdeutigkeit	Werden explizit Strategien eingesetzt, um mehrdeutige Eingaben zu klären?
17	Metaorganisation	Prompt-Komponenten (Sagar, 2024)	Lernstil festlegen („Define your Learning Style“) (Meta)	Wird eine bestimmte Darstellungsmethode der Informationen explizit verlangt?
18	Metaorganisation	Prompt Engineering (Ekin, 2023) – Techniken	Differenzierung der Aufgabenart (Meta)	Wird explizit zwischen einfachen Faktenfragen und komplexen analytischen Aufgaben unterschieden?
19	Metaorganisation	Prompt Pattern Catalog (White et al., 2023)	Meta Language Creation	Wird eine benutzerdefinierte Notation/Abkürzung eingeführt, um komplexe Sachverhalte zu vereinfachen?
20	Metaorganisation	Prompt Pattern Catalog (White et al., 2023)	Visualization Generator	Fordert der Prompt explizit eine textbasierte Visualisierungsanleitung oder visuelle Ausgabe?
21	Iteration	Prompting-Strategien (Five S, Tassoti 2024)	Share Feedback (Rückmeldung geben)	Wird gezielt um Feedback oder Korrektheit der Ergebnisse gebeten?
22	Iteration	Prompt Engineering (Ekin, 2023; Polverini & Gregorcic, 2024)	Prompt Chaining	Werden mehrschrittige, iterative Interaktionen eingesetzt?
23	Iteration	Prompt Pattern Catalog (White et al., 2023)	Flipped Interaction	Wird ChatGPT aufgefordert, selbst aktiv Fragen zu stellen, um benötigte Informationen zu sammeln?
24	allg.	Prompt Engineering (Ekin, 2023) – Best Practices	Nutzerabsicht vs. kreative Freiheiten	Ist eine Balance zwischen Nutzerintention und kreativer KI-Antwort ersichtlich?
25	allg.	Prompt Engineering (Ekin, 2023) – Best Practices	Sicherstellung ethischer Nutzung	Wird bewusst darauf geachtet, ethische Grundsätze einzuhalten?
26	Negativ		Voraussetzung von Allwissenheit	Der Prompt suggeriert, dass Open AI alles weiß, also auch das Unausgesprochene
27	Negativ		Annahme der Fähigkeit zur Fehlerdiagnose ohne ausreichende Informationen	Angabe von zu undetaillierten Informationen
28	Negativ		Übermässige Details	Der Prompt fragt zu spezifische Dinge