KI in der Programmierausbildung

Jamie Tamara Klein

Institut für Informationssicherheit und Verlässlichkeit (KASTEL)
Betreuender Mitarbeiter: Robin Maisch, M.Sc.

Diese Arbeit untersucht die Rolle und den Einfluss künstlicher Intelligenz (KI) in der modernen Programmierausbildung. Sie beleuchtet die Grundlagen der künstlichen Intelligenz, ihre vielfältigen Anwendungsfelder und konzentriert sich besonders auf KI-gestützte Tools in der Softwareentwicklung. Es wird analysiert, auf welche Art Systeme wie beispielsweise GitHub Copilot und OpenAI Codex den Programmierprozess verändern, indem sie Code-Generierung, Fehlersuche und das Erlernen neuer Programmiersprachen unterstützen. Darüber hinaus werden die Auswirkungen dieser Technologien auf die Produktivität von Entwickler:innen und die Herausforderungen bei der Integration von künstlicher Intelligenz in die Programmierausbildung diskutiert. Es werden die Chancen und Herausforderungen, die künstliche Intelligenz für Lehrende und Lernende in den Bereichen der Softwareentwicklung, Bildung und Programmierausbildung mit sich bringt, analysiert und Empfehlungen für einen verantwortungsvollen Umgang mit KI-Technologien in der Bildung herausgearbeitet.

Schlüsselwörter

Bildung, Künstliche Intelligenz, KI-Erzeugnisse generieren und detektieren, Large-Language-Models, Lehre, Programmieren, Programmier-Anfänger:innen, Programmier-Assistenten, Software-Engineering

1 Einleitung

Künstliche Intelligenz hat in den letzten Jahren in vielen Bereichen des täglichen Lebens Einzug gehalten, darunter auch in der Schule, im Beruf und im Studium. In Anbetracht dieser Entwicklung ist es empfehlenswert, einen guten Umgang mit künstlicher Intelligenz zu erlernen und zu vermitteln und gleichzeitig Methoden zu entwickeln, um KI-generierte Texte und insbesondere Code zu erkennen.

Ein besonders faszinierender Bereich, in dem künstliche Intelligenz zunehmend an Bedeutung gewinnt, ist die Softwareentwicklung und damit einhergehend die Programmierausbildung.

Die vorliegende Arbeit zielt darauf ab, einen umfassenden Überblick über den aktuellen Stand der künstlichen Intelligenz in der Programmierausbildung zu geben und Perspektiven für die Zukunft aufzuzeigen. Die Studie beginnt mit einer Darstellung der Grundlagen von künstlicher Intelligenz und ihrer verschiedenen Anwendungsfelder. Anschließend werden spezifische künstliche Intelligenzen für die Programmierung vorgestellt und ihre Einsatzmöglichkeiten in der Bildung diskutiert. Ein besonderer Fokus liegt dabei auf der Frage, wie KI-generierte Inhalte und insbesondere KI-generierter Code erkannt werden können und welche ethischen Implikationen sich daraus ergeben.

Es werden zunächst die Grundlagen der künstlichen Intelligenz betrachtet, um ein solides Verständnis für diese Technologie zu gewährleisten. Hierbei wird auch auf die verschiedenen Anwendungsfelder von künstlicher Intelligenz eingegangen, wobei das Hauptaugenmerk auf den Einsatz in der Softwareentwicklung und der Programmierausbildung gerichtet sein wird.

Ein zentraler Aspekt sind KI-gestützte Entwicklungstools wie GitHub Copilot und OpenAI Codex. Diese Systeme haben das Potenzial, den Programmierprozess grundlegend zu verändern, indem sie Entwickler:innen bei der Code-Generierung, Fehlersuche und beim Erlernen neuer Programmiersprachen unterstützen.

Darüber hinaus werden die Herausforderungen und möglichen Bedenken diskutiert, die mit dem Einsatz von künstlicher Intelligenz in der Programmierausbildung einhergehen. Dazu gehören Fragen zur Authentizität des Codes, zum tieferen Verständnis der Programmierkonzepte und zur Rolle von menschlichen Entwickler:innen in einer zunehmend KI-gestützten Umgebung.

Ziel dieser Arbeit ist es, ein umfassendes Bild der aktuellen Entwicklungen im Bereich der künstlichen Intelligenz und Programmierung zu zeichnen und Einblicke in die Chancen, Risiken und die zukünftige Gestaltung der Programmierausbildung zu geben.

2 KI als Werkzeug

In diesem Unterkapitel soll eine Übersicht gegeben werden, was künstliche Intelligenz (KI) ist, wo die Einsatzgebiete künstlicher Intelligenz liegen können und wie KI-generierte Inhalte erkannt werden können.

2.1 KI-Grundlagen

Zunächst stellt sich die Frage, was künstliche Intelligenz (KI) ist. Viele Menschen denken bei dem Begriff an moderne KI-basierte Sprachmodelle wie ChatGPT[1], doch das Thema der künstlichen Intelligenz ist schon länger präsent, wenn auch nicht in demselben Umfang wie in den letzten Jahren. Der heute noch bekannte Turing-Test wurde beispielsweise schon 1950 von Alan Turing entwickelt. Dabei handelt es sich um ein Verfahren, bei dem eine Person als Prüfer:in entscheiden muss, ob sie mit einem Menschen oder einer

[1] URL: `https://chatgpt.com/`, zuletzt aufgerufen: 20.12.2024

Maschine kommuniziert, ausschließlich basierend auf den Antworten, die die Person von dem Menschen bzw. der Maschine erhält [45].

Ein anderes Beispiel für eine frühe Form künstlicher Intelligenz ist das von 1964 bis 1966 von Joseph Weizenbaum entwickelte Computerprogramm „ELIZA“ [54]. „ELIZA“ kann Gespräche durch „Pattern Matching“ und Substitutionsmethoden simulieren, die Interaktionen mit Personen wurden durch Skripte gesteuert. Das bekannteste Skript, „DOCTOR“, simulierte eine:n Psychotherapeut:in. „ELIZA“ reagiert dabei mit nicht-direktiven Fragen auf Eingaben, basierend auf den im Skript festgelegten Regeln [54].

Künstliche Intelligenz bezieht sich grundsätzlich auf die Simulation menschlicher Intelligenz in Maschinen, die programmiert sind, um wie Menschen zu denken und zu handeln [11]. Zu KI-Technologien gehören unter anderem maschinelles Lernen, bei dem Computer aus vergangenen Erfahrungen lernen, und „Deep Learning“, das neuronale Netzwerke zur Verarbeitung komplexer Muster nutzt [11].

Ein wichtiger Bereich innerhalb des Feldes der künstlichen Intelligenz ist auch Natural Language Processing (NLP), der sich mit der Kommunikation von Computern und Menschen mithilfe von menschlicher Sprache beschäftigt. NLP-Technologien ermöglichen es Maschinen, Texte zu verstehen, zu interpretieren und zu generieren, was für die Entwicklung großer Sprachmodelle entscheidend ist [42].

Solche großen Sprachmodelle (engl. „Large Language Models“, LLMs), zu denen unter anderem ChatGPT zählt, sind eine bedeutende Untergruppe des maschinellen Lernens [38]. Solche LLMs sind in der Lage, menschenähnliche Texte zu generieren und zu verstehen und Aufgaben wie Sprachübersetzung zu übernehmen. Diese Modelle haben in der breiten Öffentlichkeit an Bekanntheit gewonnen, insbesondere durch ihre einfache Anwendung [38].

Im folgenden Abschnitt wird weiter auf den aktuellen Einsatz künstlicher Intelligenz eingegangen und wie sie Teil des täglichen Lebens geworden sind.

2.2 Anwendungsfelder von KI

In diesem Abschnitt sollen die verschiedenen Anwendungsfelder künstlicher Intelligenz beleuchtet werden.

In der Industrie steuert künstliche Intelligenz Produktionsprozesse, lenkt Warenströme und sagt drohende Maschinenausfälle voraus. „Predictive Maintenance“ nutzt künstliche Intelligenz, um den wahrscheinlichsten Zeitpunkt für Wartungsarbeiten zu ermitteln und die Planung zu optimieren [31]. Künstliche Intelligenz wird auch zur Verbesserung der Prozessqualität, zur Optimierung des Energieverbrauchs und für „Predictive Analytics“ eingesetzt [41].

Auch im Transportwesen spielt künstliche Intelligenz eine zunehmend wichtigere Rolle, insbesondere im Bereich des autonomen Fahrens. KI-Systeme können heutzutage komplexe Verkehrssituationen erkennen und die Wahrscheinlichkeit für Unfälle reduzieren [65]. In Zukunft könnte künstliche Intelligenz zusätzlich vermehrt zur Verbesserung der Verkehrssteuerung in Städten und auf Autobahnen genutzt werden und so zur Reduzierung von Staus beitragen [55].

In der Landwirtschaft könnte künstliche Intelligenz nicht nur für das autonome Fahren verschiedener Maschinen genutzt werden, sondern vor allem zur Optimierung von

Anbaumethoden, zur Überwachung der Gesundheit der Pflanzen und zur besseren Vorhersage von Ernteerträgen. KI-gestützte Systeme können Daten von Satelliten, Drohnen und Bodensensoren analysieren, um genaue Empfehlungen für Bewässerung, Düngung und Pflanzenschutz zu geben oder gar selbst durchzuführen [46]. Zukünftig könnte künstliche Intelligenz auch verstärkt in der Entwicklung widerstandsfähigerer Pflanzensorten und in der Automatisierung weiterer landwirtschaftlicher Prozesse zum Einsatz kommen [37].

Eine solche nachhaltig ausgerichtete Landwirtschaft ist eng mit dem Thema Umweltschutz verknüpft. KI-Systeme, die in der Landwirtschaft eingesetzt werden, liefern wertvolle Daten für die Klimaforschung. Sie können zur Analyse von Klimadaten, zur Vorhersage von Wetterextremen und zur Optimierung von Ressourcennutzung eingesetzt werden [20].

Die Erkenntnisse aus der KI-gestützten Klimaforschung fließen direkt in die Vorhersage und Bewältigung von Naturkatastrophen ein. Dort können sie bei der Verbesserung von Frühwarnsystemen helfen, die Verteilung von Ressourcen optimieren und bei der optimalen Koordination von Rettungseinsätzen in Katastrophengebieten unterstützen [22].

Künstliche Intelligenz kann nicht nur global, sondern auch lokal bei Einzelschicksalen eine Hilfe sein, beispielsweise als Unterstützung im Bereich der Medizin. In der Medizin hat künstliche Intelligenz enormes Potenzial zur Verbesserung der Diagnostik und Behandlung. Machine-Learning-Algorithmen können Tumore oder Schlaganfälle auf der Basis von CT-Scans erkennen oder Hautveränderungen klassifizieren [56, 40]. Künstliche Intelligenz unterstützt das medizinische Personal bei der Erstellung genauerer Diagnosen auch schon in einem früheren Stadium des Krankheitsverlaufs und hilft bei der Entwicklung von individuellen Behandlungsplänen. Zukünftig könnte künstliche Intelligenz auch in der Arzneimittelforschung eine größere Rolle spielen, indem sie die Entwicklung neuer Medikamente beschleunigt und optimiert [62].

Ein anderes Einsatzgebiet für künstliche Intelligenz ist der Finanzsektor, hier wird künstliche Intelligenz für verschiedene Zwecke eingesetzt. Künstliche Intelligenz kann bei Betrugserkennung, Themen digitaler Sicherheit und Datensicherheit zum Einsatz kommen [40]. Hier können KI-gestützte Analysen von in der Vergangenheit aufgezeichneten Daten und Echtzeitdaten genutzt werden, um schnellere Entscheidungen über die Durchführung von Transaktionen treffen zu können. Darüber hinaus nutzen Finanzdienstleister:innen künstliche Intelligenz für Entscheidungen bezüglich Vermögensverwaltung, Kreditgenehmigungen und Handelsentscheidungen [23].

Damit verwandt sind diverse Sicherheitsbereiche – dort wird künstliche Intelligenz zur Erkennung von Bedrohungen und zur Verbesserung von Überwachungssystemen eingesetzt. KI-gestützte Systeme können Anomalien im Netzwerkverkehr oder Benutzer:innenverhalten erkennen und so potenzielle Sicherheitsrisiken frühzeitig identifizieren [9]. In der physischen Sicherheit werden KI-Algorithmen zur Gesichtserkennung und Verhaltensanalyse eingesetzt – beispielsweise kann eine künstliche Intelligenz erkennen, ob eine Person in einem Geschäft versucht, ein Produkt zu stehlen [33]. Zukünftig könnte künstliche Intelligenz auch eine größere Rolle in der Entwicklung fortschrittlicher Verschlüsselungstechniken und in der Abwehr von Cyberangriffen spielen, sowie im Bereich der physischen Sicherheit vermehrt zum Einsatz kommen [9, 33].

Künstliche Intelligenz beeinflusst auch den kreativen und unterhaltenden Sektor auf vielfältige Art und Weise. Im Unterhaltungsbereich wird künstliche Intelligenz zur Per-

sonalisierung von Inhalten und zur Verbesserung von Empfehlungssystemen eingesetzt. Zum Beispiel nutzen Streaming-Dienste KI-Algorithmen, um Nutzer:innen personalisierte Inhalte vorzuschlagen [44]. Darüber hinaus findet künstliche Intelligenz Anwendung in der Filmindustrie für visuelle Effekte und in der Musikindustrie für die Komposition und Produktion von Musik [15]. Die Technologie wird zunehmend zur Generierung von Musik, zur Erstellung von Kunstwerken und zur Unterstützung bei der Filmproduktion eingesetzt [51]. In Zukunft könnte künstliche Intelligenz auch verstärkt in der Entwicklung interaktiver und immersiver Unterhaltungsformate zum Einsatz kommen [44].

Zu diesem Bereich kann man auch die KI-gestützte Textgenerierung zählen, die in den letzten Jahren bemerkenswerte Fortschritte gemacht hat. Künstliche Intelligenzen wie zum Beispiel ChatGPT oder *HuggingChat*[2] können mit Eingabeaufforderungen als Grundlage komplexe und vielfältige Texte erzeugen [34]. Diese Technologie findet Anwendung in der Erstellung von vielseitigen Inhalten, der Beantwortung von verschiedensten Fragen und der Unterstützung bei kreativen Schreibprozessen [63]. Zukünftig könnte die KI-gestützte Textgenerierung in der Erstellung von personalisierten Nachrichten, der Entwicklung von Büchern, Poesie, Drehbüchern oder weiteren Erzeugnissen in Textform wie der automatisierten Berichterstattung eine große Rolle spielen [34].

Auch die KI-gestützte Bildgenerierung hat in den letzten Jahren enorme Fortschritte gemacht. Systeme wie *DALL-E*[3], *Midjourney*[4] und *Stable Diffusion*[5] können auf Basis von Beschreibungen in Textform hochwertige Bilder erzeugen [26]. Diese Technologie findet Anwendung in der Kunst, im Design und in der Werbung [34]. Zukünftig könnte die KI-gestützte Bildgenerierung auch in der Produktentwicklung, in der virtuellen Realität und in der Erstellung von Lehrmaterialien eine größere Rolle spielen [24].

Im Bereich der Computerspiele verbessert künstliche Intelligenz neben dem teilweisen Einsatz bei der Erstellung von Bild und Ton das Spielerlebnis und schafft realistischere Szenarien. KI-gesteuerte Non-Player-Charactere (NPCs) zeigen komplexere Verhaltensweisen und Reaktionen, was zu einer immersiveren Spielerfahrung führt [18]. Zudem werden KI-Algorithmen zur Generierung prozeduraler Inhalte wie Landschaften oder Quests verwendet, was die Vielfalt und Wiederbespielbarkeit von Spielen erhöht [28].

Ein weiterer Fortschritt im Bereich der KI-Assistenten ist *Microsoft Copilot*[6], der in verschiedene *Microsoft*-Produkte integriert wird. Dieser KI-gestützte Assistent nutzt die Leistungsfähigkeit von Large Language Models, um Nutzer:innen bei einer Vielzahl von Aufgaben zu unterstützen [8]. In Unternehmen wird Microsoft Copilot beispielsweise eingesetzt, um E-Mails zu verfassen, Präsentationen zu erstellen, Dokumente zu finden oder Daten in Excel zu analysieren. Darüber hinaus setzen einige Unternehmen auf personalisierte ChatGPT-Instanzen, die auf ihre spezifischen Bedürfnisse ausgerichtet sind. Diese maßgeschneiderten KI-Assistenten ermöglichen es Unternehmen verschiedenster Art, die Vorteile von KI zu nutzen, während sie gleichzeitig die Kontrolle über sensible Daten behalten und spezifische Branchenkenntnisse in die Systeme integrieren [8].

[2]URL: `https://huggingface.co/chat/`, zuletzt aufgerufen: 20.12.2024
[3]URL: `https://openai.com/index/dall-e-3/`, zuletzt aufgerufen: 20.12.2024
[4]URL: `https://www.midjourney.com`, zuletzt aufgerufen: 20.12.2024
[5]URL: `https://stablediffusion.com/de`, zuletzt aufgerufen: 20.12.2024
[6]URL: `https://copilot.microsoft.com/`, zuletzt aufgerufen: 20.12.2024

Ein wichtiger Bereich, in dem künstliche Intelligenz Anwendung finden kann, ist in der Bildung von Kindern, Jugendlichen und auch Erwachsenen. Im Bildungssektor wird künstliche Intelligenz zur Personalisierung von Lernprozessen eingesetzt. Adaptive Lernsysteme können den Fortschritt und die Bedürfnisse einzelner Schüler:innen analysieren und den Unterricht entsprechend anpassen [12]. KI-gestützte Tutorien-Systeme können mühelos individuell zugeschnittene Unterstützung bieten und Lehrkräften bei der Bewertung und Korrektur von Aufgaben helfen [66]. In Zukunft könnte künstliche Intelligenz auch zur Entwicklung innovativer Lehrmethoden und zur Verbesserung der Zugänglichkeit von Bildung beitragen [12].

In der Forschung sind ebenfalls KI-Technologien zu finden, die die Arbeit von wissenschaftlichen Mitarbeitenden erleichtern können. Beispielsweise *Research Rabbit*[7], eine KI-basierte Plattform, die Wissenschaftler:innen bei der Literaturrecherche unterstützt. Sie nutzt Machine-Learning-Algorithmen, um relevante Artikel zu identifizieren und visualisiert Verbindungen zwischen Referenzarbeiten. Eine teilweise ähnliche Plattform ist *ScienceOS*[8], sie dient ebenfalls der Optimierung des wissenschaftlichen Forschungsprozesses und bietet KI-basierte Werkzeuge zur Literaturrecherche und Datenanalyse [43].

Auch im Bereich der Software-Entwicklung und bei der Vermittlung von Programmierkenntnissen sind Veränderungen durch KI-gestützte Werkzeuge bzw. Lernwerkzeuge zu beobachten. Im Bereich der Software-Entwicklung werden KI-Systeme zunehmend zur Unterstützung von Programmierer:innen eingesetzt. Künstliche Intelligenzen wie *GitHub Copilot*[9] und *OpenAI Codex* [10] können auf Basis von natürlichsprachlichen Beschreibungen oder bereits vorhandenem Code neue Codevorschläge generieren [40]. Diese Systeme können die Produktivität von Entwickler:innen steigern, indem sie Code generieren oder bei der Fehlersuche in bestehendem Code helfen und den Einstieg in neue Programmiersprachen erleichtern [47]. Auch beim Testen von Software können KI-Anwendungen von Nutzen sein [40].

Insgesamt zeigt sich, dass künstliche Intelligenz in nahezu allen Bereichen unseres Lebens und der Wirtschaft Einzug gehalten hat und sehr wahrscheinlich weiter an Bedeutung gewinnen wird. Die kontinuierliche Weiterentwicklung von KI-Technologien verspricht, in Zukunft noch leistungsfähigere und vielseitigere Anwendungen hervorzubringen, die unser Leben und Arbeiten grundlegend verändern werden.

2.3 Erkennen von KI-generiertem Inhalt

Nachdem die Möglichkeiten betrachtet wurden, in welchen Bereichen künstliche Intelligenz genutzt werden kann und welche Inhalte sie erzeugen kann, ist es empfehlenswert zu betrachten, wie solche KI-generierten Inhalte erkannt und von menschlich generierten Inhalten unterschieden werden können. Die Erkennung von KI-generierten Texten, Bildern und Code-Abschnitten ist ein wichtiges Thema, sowohl in der Forschung als auch in

[7] URL: `https://researchrabbitapp.com/`, zuletzt aufgerufen: 20.12.2024
[8] URL: `https://www.scienceos.ai/`, zuletzt aufgerufen: 20.12.2024
[9] URL: `https://github.com/features/copilot`, zuletzt aufgerufen: 20.12.2024
[10] URL: `https://openai.com/index/openai-codex/`, zuletzt aufgerufen: 20.12.2024

der Kunst. Dabei geht es insbesondere um die Integrität von Informationen, den Schutz geistigen Eigentums und die Authentizität digitaler Inhalte [7, 10].

KI-generierte Texte können oft durch bestimmte Merkmale identifiziert werden. Menschliche Betrachter:innen können auf subtile Hinweise in der Schreibweise, dem Stil und der Struktur achten, um KI-generierte Texte zu erkennen. KI-Modelle neigen dazu, bestimmte Muster zu wiederholen oder besonders ausschweifende Formulierungen zu nutzen [30]. Außerdem können sie Schwierigkeiten haben, emotionale Nuancen oder kulturelle Kontexte darzustellen [10, 21]. Für die automatisierte Erkennung von durch künstliche Intelligenz erzeugter Texte stehen verschiedene Werkzeuge zur Verfügung. Ein Beispiel ist *ZeroGPT*[11], ein kostenloser Online-Dienst, der wiederum eine künstliche Intelligenz verwendet, um zu analysieren, ob ein Text von künstlicher Intelligenz wie ChatGPT oder von einem Menschen generiert wurde [61, 27].

In Anbetracht von Deepfakes und manipulierten Fotos, mit denen *Fakenews* glaubhaft gemacht werden können, ist es wichtiger denn je, KI-generierte Bilder zuverlässig erkennen zu können [25]. *Illuminarty*[12] ist eine beispielhafte Plattform, die darauf spezialisiert ist, KI-generierte Bilder zu erkennen. Sie kombiniert verschiedene Bildanalyse-Verfahren und berechnet die Wahrscheinlichkeit, dass ein Bild mit einem KI-Bildgenerator erstellt wurde. Zusätzlich bietet Illuminarty eine Aufschlüsselung nach verwendetem KI-Bildgenerator und kann spezifische Regionen im Bild identifizieren, die auf eine KI-Generierung hindeuten. Menschen können KI-generierte Bilder teilweise erkennen, indem Details genauer betrachtet werden: Eine falsche Anzahl an Fingern einer Hand, eine unstimmige Darstellung von Haaren, eine falsche Anzahl an Beinen bei einem Tier oder unstimmige Körperproportionen können Anzeichen sein, dass ein Bild von künstlicher Intelligenz erzeugt wurde [60, 3].

Die Erkennung von KI-generiertem Code stellt eine besondere Herausforderung dar, da Code oft strukturierte und wiederholbare Muster aufweist. KI-Code-Detektoren sind spezielle Softwareanwendungen, die Algorithmen der künstlichen Intelligenz verwenden, um Code zu analysieren. *Codequiry*[13] ist ein Beispiel für KI-Code-Prüfprogramme. Es analysiert Codeausschnitte und hebt von künstlicher Intelligenz geschriebenen Code hervor, um ihn vom menschlich geschriebenen Code zu unterscheiden. Solche Werkzeuge, die in Unterabschnitt 3.2 genauer analysiert werden, untersuchen Codierungsmuster und die Komplexität innerhalb des Codierungsstils, um zu bewerten, wie wahrscheinlich es ist, dass künstliche Intelligenz den Code erzeugt hat [61]. Es ist wichtig zu beachten, dass sich die Erkennung von KI-generiertem Code oft schwierig gestaltet und eine gewisse Expertise erfordert – sowohl vom Menschen als auch von der Maschine [7]. Die Kombination aus automatisierten Werkzeugen und menschlichem Urteilsvermögen bleibt vorerst der effektivste Ansatz zur Identifizierung von KI-generiertem Code [61]. Im folgenden Kapitel wird noch einmal genauer auf die Möglichkeiten eingegangen, wie KI-generierter Code erkannt werden kann und beleuchtet, wie gut technische Hilfsmittel dafür geeignet sind.

Die Erkennung von KI-generierten Inhalten ist eine Aufgabe, die sowohl menschliche Intuition als auch technologische Unterstützung erfordert. Während Menschen oft subtile

[11]URL: `https://zerogpt.net/de`, zuletzt aufgerufen: 20.12.2024
[12]URL: `https://illuminarty.ai/de/`, zuletzt aufgerufen: 20.12.2024
[13]URL: `https://codequiry.com/`, zuletzt aufgerufen: 20.12.2024

stilistische Merkmale erkennen können, bieten moderne Werkzeuge, die ebenfalls auf künstlicher Intelligenz beruhen, eine systematische und datenbasierte Herangehensweise. Die Kombination aus menschlichem Urteil und maschineller Analyse ist ein vielversprechender Ansatz, um von künstlicher Intelligenz erzeugte Inhalte als solche erkennen zu können.

3 KI-generierter Code

In den letzten Jahren haben sich verschiedene KI-Modelle entwickelt, die sowohl angehende als auch etablierte und erfahrene Programmierer:innen bei ihrer Arbeit unterstützen können. Diese künstlichen Intelligenzen bieten unterschiedliche Funktionen und eignen sich für verschiedene Zielgruppen sowie Anwendungsfälle. Einige künstliche Intelligenzen werden hier in einem kurzen Überblick vorgestellt.

OpenAI Codex ist ein KI-Modell, das natürliche Sprache in Programmcode übersetzen kann. Es basiert auf dem GPT-3 Sprachmodell, wurde aber speziell für die Codegenerierung optimiert. Codex beherrscht diverse Programmiersprachen, darunter Python, JavaScript, Go, Perl, PHP, Ruby, Swift und TypeScript [19].

GitHub Copilot basiert auf OpenAI's Codex-Modell und ist eines der bekanntesten KI-Werkzeuge für Entwickler:innen. Es bietet Funktionen wie Code-Vervollständigung, Fehlerbehebung und Code-Erklärungen verschiedensten Programmiersprachen [59]. Copilot hat sich als besonders nützlich für erfahrene Entwickler:innen erwiesen, da es die Produktivität steigern und repetitive Aufgaben beschleunigen kann [36].

Amazon's CodeWhisperer ist ein weiteres leistungsfähiges Werkzeug, das sich nahtlos in verschiedene Entwicklungsumgebungen integriert. Es bietet intelligente Code-Vorschläge, automatische Dokumentationsgenerierung und Fehlererkennung. CodeWhisperer ist besonders nützlich für Entwickler:innen, die mit AWS-Diensten arbeiten, da es spezifische Kenntnisse in diesem Bereich einbringt [64].

Google's Bard ist weniger auf Codegeneration spezialisiert als andere Werkzeuge, aber bietet Unterstützung bei verschiedenen Programmier- und Softwareentwicklungsaufgaben. Es kann bei der Codegenerierung, Fehlerbehebung und Code-Erklärung in über 20 Programmiersprachen helfen [5].

Tabnine ist ein KI-gestützter Code-Assistent, der sich auf Codeergänzungen spezialisiert hat. Es lernt aus der eigenen Codebasis und passt Vorschläge entsprechend an. Tabnine ist besonders nützlich für Entwickler:innen, die in größeren Teams oder an umfangreichen Projekten arbeiten, da es projektspezifische Vorschläge machen kann [53].

DeepMind's AlphaCode hat in standardisierten Programmierwettbewerben beeindruckende Ergebnisse erzielt und kann mit durchschnittlichen menschlichen Programmierern mithalten. Obwohl AlphaCode nicht für Endnutzer:innen verfügbar ist, zeigt es das Potenzial von künstlicher Intelligenz im Bereich der algorithmischen Problemlösung [35].

3.1 KI-generierter Code in der Praxis

Im professionellen Umfeld werden KI-Programmierassistenten für eine Vielzahl von Aufgaben genutzt, wie Liang, Yang und Myers [36] dargelegt haben. Dazu haben sie 410

GitHub-Nutzer:innen zu ihrem Nutzungsverhalten von künstlicher Intelligenz beim Programmieren befragt, welche künstlichen Intelligenzen sie verwenden und wie oft und was für oder gegen eine solche Nutzung sprechen.

In der Studie von Liang, Yang und Myers [36] wurden die Befragten zu ihrer Nutzung von Codegenerierungstools befragt. Demnach gaben 306 Teilnehmende an, GitHub Copilot zu verwenden, während 118 TabNine und 50 Amazon CodeWhisperer nutzen. Zudem berichteten 25 Befragte von der Verwendung von ChatGPT, und 54 Teilnehmende setzten organisationsspezifische KI-Werkzeuge ein, die auf proprietärem Code trainiert wurden. Dabei ist anzumerken, dass einige der Befragten nicht nur eine, sondern mehrere künstliche Intelligenzen beim Programmieren verwenden.

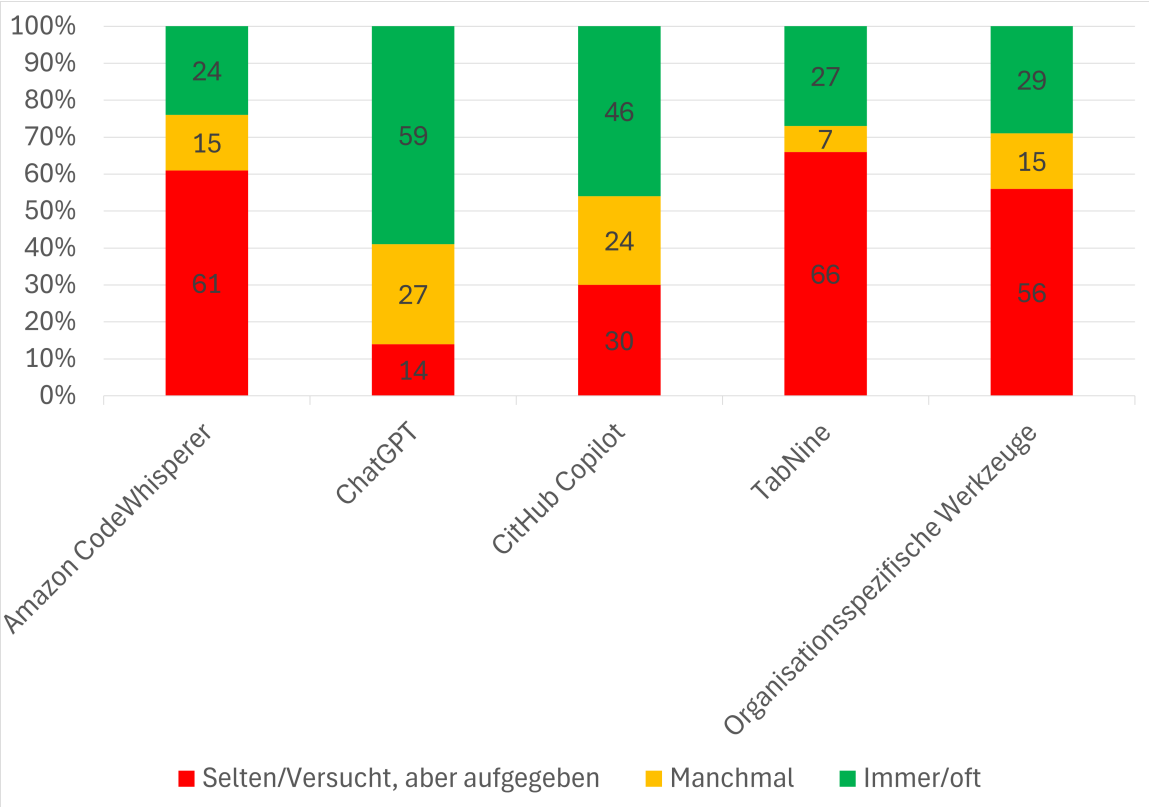


Abbildung 1: Nutzungsverhalten der Studienteilnehmenden, nach [36]

In Abbildung 1 ist das Nutzungsverhalten der Befragten zu sehen. Das am meisten genutzte GitHub Copilot wird von 46 Prozent der Befragten „immer“ oder „oft“ genutzt und von 24 Prozent „manchmal“. Das am zweithäufigsten genutzte Werkzeug TabNine wird im Gegensatz dazu nur von 27 Prozent bzw. 7 Prozent „immer/oft“ bzw. „manchmal“ genutzt und von 66 Prozent „selten“ oder sie haben es versucht, aber aufgegeben. Ein ähnliches Bild zeichnet sich auch bei der Verteilung des Nutzungsverhaltens von Amazon CodeWhisperer und den organisationsspezifischen KI-Werkzeugen ab, hier nutzen 24 Prozent bzw. 29 Prozent „immer/oft“, aber auch 61 Prozent bzw. 56 Prozent „selten“ oder haben die Nutzung aufgegeben. In dieser Statistik überzeugt vor allem ChatGPT, das in

59 Prozent der Fälle „immer/oft“ genutzt wird und nur von 14 Prozent der Befragten nur „selten“ genutzt wird bzw. ausprobiert, aber nicht mehr genutzt wird.

Relevant ist außerdem, wie viel Prozent des geschriebenen Codes tatsächlich von den künstlichen Intelligenzen stammt. Hier erreichen die organisationsspezifischen KI-Werkzeuge den höchsten Wert von 37 Prozent, gefolgt von GitHub Copilot mit 30,5 Prozent. Obwohl ChatGPT den größten Anteil an häufigen Nutzer:innen hat, erreicht es wie TabNine nur einen Wert von 20 Prozent des geschriebenen Codes. Nur Amazon CodeWhisperer erreicht mit 5 Prozent einen geringeren Wert.

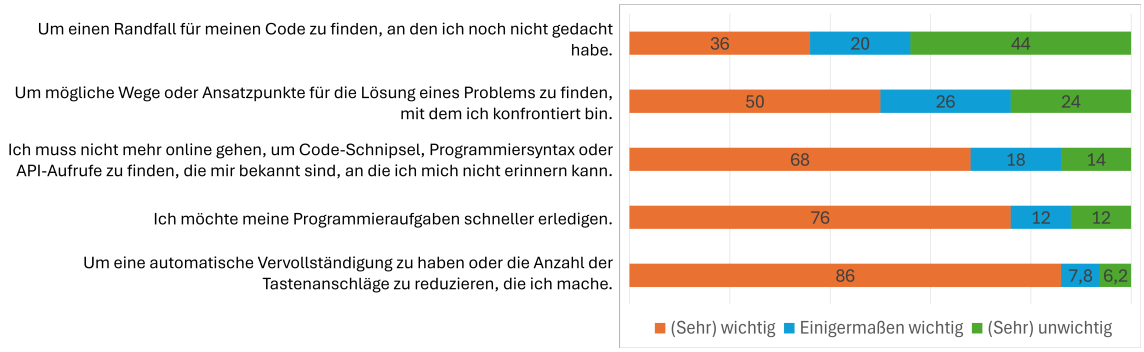


Abbildung 2: Gründe der Teilnehmenden, KI beim Programmieren zu verwenden, nach [36]

Liang, Yang und Myers [36] untersuchten außerdem die Gründe, die die Befragten angaben, die für und gegen die Nutzung von künstlicher Intelligenz beim Programmieren sprechen. Die Gründe für eine Nutzung sind in Abbildung 2 abzulesen. Vor allem Gründe der Bequemlichkeit beim Programmieren spielten für die Befragten eine sehr wichtige oder wichtige Rolle. Dazu gehören Autovervollständigung mit 86 Prozent, schnelleres Erledigen von Aufgaben mit 76 Prozent und die Ersparnis einer Online-Suche, weil man sich nicht mehr an Code-Abschnitte, Syntax oder API-Aufrufe erinnern kann, mit 68 Prozent. 50 Prozent der Befragten gaben außerdem an, künstliche Intelligenz beim Programmieren zu verwenden, um Lösungsansätze zu finden. 36 Prozent nutzen die KI-Werkzeuge auch für das Finden von Randfällen in Programmen.

Einige Teilnehmenden schienen die KI-Werkzeuge jedoch nicht oder nur selten zu verwenden, insbesondere, da diese keine nützlichen oder relevanten Ergebnisse lieferten. Diese und weitere Begründungen sind in Abbildung 3 zu sehen. Ein wichtiger Grund war, dass die generierten Programmabschnitte die funktionalen oder nicht-funktionalen Anforderungen nicht erfüllten. 54 Prozent der Befragten gaben dies als sehr wichtigen bzw. wichtigen Grund an. Darüber hinaus sahen 48 Prozent der Teilnehmenden einen sehr wichtigen bzw. wichtigen Grund darin, dass sie Schwierigkeiten hatten, die Werkzeuge zu kontrollieren, um den gewünschten Code zu erhalten. Außerdem mussten die Befragten ihrem Ermessen nach zu viel Zeit mit dem Korrigieren oder Anpassen des Codes verbringen, was 38 Prozent als problematisch empfanden.

Ein weiterer wichtiger Grund für die Ablehnung gegenüber den künstlichen Intelligenzen war, dass sie häufig keine hilfreichen Vorschläge lieferten – 34 Prozent der Teilnehmenden betrachteten das als einen sehr wichtigen bzw. wichtigen Grund. Darüber

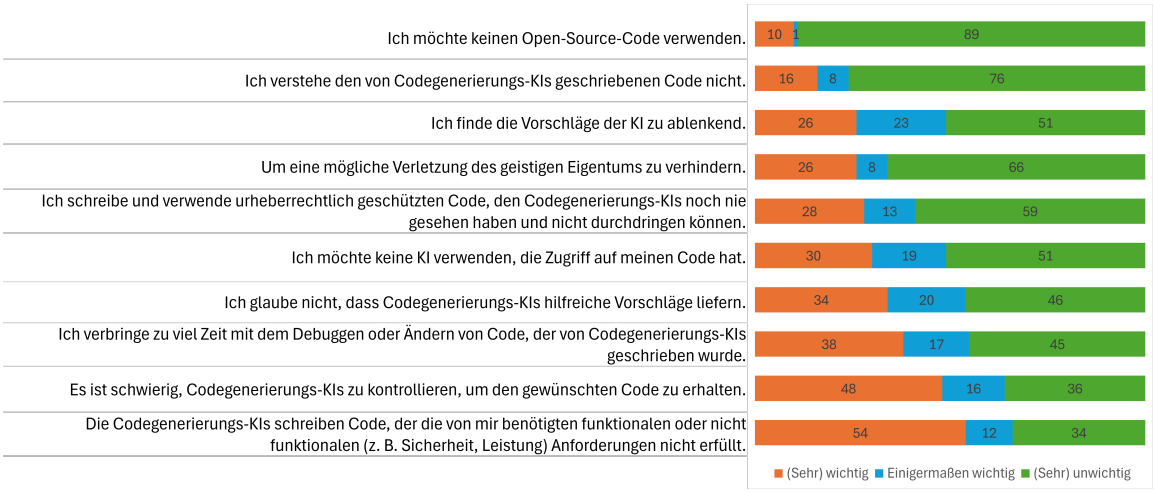


Abbildung 3: Gründe der Teilnehmenden, KI beim Programmieren nicht zu verwenden, nach [36]

hinaus gaben 30 Prozent der Befragten an, dass sie künstliche Intelligenzen, die Zugriff auf ihren Code haben, ablehnen.

Ähnliche Gründe sahen auch 28 Prozent bzw. 26 Prozent der Befragten als sehr wichtig oder wichtig an: Die Arbeit mit eigenem urheberrechtlich geschütztem Code, den die künstliche Intelligenz noch nicht kennt und nicht ausreichend gut verarbeiten kann, und Angst vor einer Verletzung des geistigen Eigentums.

Einige Teilnehmende empfanden es auch sehr wichtig bzw. wichtig, dass die Vorschläge der Tools auch zu ablenkend sind (26 Prozent), während 16 Prozent angaben, den von den Tools generierten Code nicht zu verstehen, was ebenfalls als ein wesentliches Hindernis wahrgenommen wurde. Schließlich gaben 10 Prozent der Befragten an, dass sie keinen Open-Source-Code verwenden möchten.

Wenn die Befragten die KI-Werkzeuge erfolgreich einsetzen konnten, war die Generierung von repetitivem Code der häufigste Anwendungsfall, dies wurde von 78 der 410 Teilnehmenden genannt. Entwickler:innen nutzen KI-Werkzeuge nicht nur zur Generierung von repetitiven Code, sondern auch erfolgreich für die Erstellung von „Boilerplate“-Code oder häufig vorkommenden Programmieraufgaben in Studienarbeiten. Ein:e Teilnehmer:in betonte beispielsweise, dass künstliche Intelligenz besonders nützlich ist für Code, der hochgradig repetitiv ist, aber nicht direkt kopiert und eingefügt werden kann. Der zweithäufigste Einsatzbereich mit 68 Nennungen ist die Generierung von Code mit einfacher Logik, wie kleinere Hilfsfunktionen. Code-Vervollständigung wurde noch 28 Mal als Anwendungsfall genannt, hier nutzen Entwickler:innen die künstlichen Intelligenzen für kurze Autovervollständigungen, was die Programmiergeschwindigkeit erhöht. Im Bereich Qualitätssicherung, der 21 Mal als Anwendungsfall genannt wurde, werden künstliche Intelligenzen zur Generierung von sinnvollen Protokollmeldungen und zur schnellen Erstellung von Testfällen eingesetzt. Besonders wertvoll scheint zu sein, dass Künstliche Intelligenz auch für unwahrscheinliche Szenarien Testfälle erstellt. Künstliche Intelligenzen unterstützen auch beim Erstellen von „Proof-of-Concepts“ (20 Nennungen), indem sie mehrere Implementierungen für ein Problem generieren. Sie dienen oft als

Ausgangspunkt, wenn Entwickler:innen eine neue Lösung benötigen oder nur eine vage Idee haben.

Beim Erlernen neuer Programmiersprachen oder Bibliotheken, das von 19 Teilnehmenden genannt wurde, ersetzen KI-Werkzeuge oft Lernmethoden wie Online-Dokumentationen oder Video-Tutorials. Hier ist zu beachten, dass es sich nicht um komplette Anfänger:innen handelt, sondern um Personen, die schon programmieren können und zusätzlich den Umgang mit einer unbekannten Bibliothek oder eine neue Programmiersprache lernen, beispielsweise für ein mehrsprachiges Projekt. Ebenso häufig werden KI-Assistenten genutzt, um sich an vergessene Syntax oder API-Methoden zu erinnern, was die Notwendigkeit von Web-Suchen reduziert.

Effizienzsteigerung wurde 18 Mal erwähnt. Künstliche Intelligenzen können Entwickler:innen helfen, im Arbeitsfluss zu bleiben und Unterbrechungen zu vermeiden. Einige Teilnehmende nutzten KI-Assistenten auch zur Generierung von Dokumentation (6 Nennungen), was die Zusammenarbeit im Team verbessern kann. Schließlich wurden die künstlichen Intelligenzen auch zur Verbesserung der Code-Konsistenz eingesetzt (4 Nennungen), um einheitlichen Codestil und korrekte Einrückungen in verschiedenen Programmiersprachen sicherzustellen.

Dieselben Teilnehmenden wurden im Rahmen dieser Studie auch gefragt, mit welchen Schwierigkeiten sie bei der Benutzung der künstlichen Intelligenzen konfrontiert sind.

Die größten Schwierigkeiten, die Entwickler:innen häufig erlebten, waren:

- Mangelnde Nachvollziehbarkeit (30 Prozent der Teilnehmenden)
  Entwickler:innen konnten oft nicht erkennen, welcher Teil ihrer Eingabe die Ausgabe des KI-Assistenten beeinflusst hatte.

- Aufgeben bei der Nutzung des generierten Codes (28 Prozent der Teilnehmenden)
  Viele Entwickler:innen gaben die Verwendung des KI-generierten Codes auf.

- Kontrollprobleme (26 Prozent der Teilnehmenden)
  Entwickler:innen hatten Schwierigkeiten, das KI-Modell zu steuern, um die gewünschten Ergebnisse zu erzielen.

Nur 5,6 Prozent der Teilnehmenden hatten Probleme damit, den generierten Code zu verstehen, obwohl dies in früheren Studien als Problem identifiziert wurde [57]. Bei der Modifikation des generierten Codes berichteten 63 Prozent der Teilnehmenden der Studie von Liang, Yang und Myers [36] von regelmäßigem Erfolg. Die meisten (62 Prozent) änderten den generierten Code direkt, während 40 Prozent den Eingabekontext anpassten. 44 Prozent der Teilnehmenden verwendeten den generierten Code häufig unverändert.

Manche der Teilnehmenden entwickelten bestimmte Strategien, um die Ergebnisse der künstlichen Intelligenzen zu verbessern. Die häufigste Strategie mit 99 Nennungen war, dass der künstlichen Intelligenz klare und detaillierte Erklärungen zum zu erstellenden Code gegeben wurden. 36 Teilnehmende boten der künstlichen Intelligenz zusätzlichen Code, um ihr einen Anhaltspunkt zu geben. Andere Teilnehmende (24) orientierten sich an gängigen Konventionen, wie den Regeln von Entwickler:innengemeinschaften, Designmustern oder der Verwendung von gut benannten Variablen. Je 18 Nennungen gab es für die Strategien, die Anweisungen an die künstliche Intelligenz aufzuspalten und

der künstlichen Intelligenz schon vorhandenen Code zur Verfügung zu stellen, um sie mit Kontext zu versorgen. 13 Teilnehmende gaben an, ihre Eingaben iterativ zu ändern, um die Ausgaben der künstlichen Intelligenz zu optimieren (z. B. Umformulieren der Eingabeaufforderungen). Es gab allerdings auch 44 Nennungen, dass keine bestimmte Strategie angewendet wird, da das Ergebnis der KI-Programmierassistenz als ausreichend gut angesehen wird.

Bei der Konfrontation mit den zuletzt genannten Problemen und dem Umgang damit zeigt sich auch ein Unterschied zwischen den in der Studie von Liang, Yang und Myers [36] befragten erfahrenen Programmierer:innen und Menschen, die wenig Erfahrung im Programmieren haben. Choudhuri u. a. [13] fanden heraus, dass Student:innen, die Programmieraufgaben mithilfe von ChatGPT (GPT-4) bearbeitet haben, etwa so gute Ergebnisse erzielen wie Student:innen, denen keine künstliche Intelligenz zur Verfügung stand. Allerdings war die Frustration bei der Gruppe, die ChatGPT genutzt hat, deutlich höher als bei der anderen Gruppe. Das lag insbesondere daran, dass ChatGPT nur begrenztes Wissen in Nischen-Bereichen hat, das Problem teilweise nicht verstanden hat, halluziniert oder falsche Anweisungen gibt. Auch wenn ChatGPT die Frage verstanden hat und eine sinnvolle Ausgabe liefert, kam es vor, dass die Lösung nicht vollständig war oder nur teilweise korrekt [13].

Von den Teilnehmenden, die ChatGPT zur Bearbeitung der Aufgaben genutzt haben, gab es gemischte Meinungen, ob sie ChatGPT zukünftig zum Lernen oder Programmieren ähnlicher Aufgaben nutzen möchten und ob sie es weiterempfehlen würden. Diese Teilnehmenden gaben außerdem an, ob und wenn ja, wie stark ChatGPT von den Richtlinien für Mensch-KI-Interaktion, die von Microsoft entwickelt wurden, abweicht [2, 13]. Die von Choudhuri u. a. [13] betrachteten Richtlinien sind:

- G1: Deutlich machen, was das System tun kann
- G2: Verdeutlichen, wie gut das System tun kann, was es kann
- G4: System zeigt kontextuell relevante Informationen
- G5: Entspricht relevanten sozialen Normen
- G6: Soziale Voreingenommenheit abschwächen
- G7: Effizientes Aufrufen unterstützen
- G8: Effiziente Entlassung unterstützen
- G9: Effiziente Korrektur unterstützen
- G10: Leistungen im Zweifelsfall ausweiten
- G11: Verdeutlichen, warum das System getan hat, was es getan hat
- G12: System erinnert sich an die letzten Interaktionen
- G13: Aus Benutzer:innenverhalten lernen

- G14: Vorsichtiges Aktualisieren und Anpassen
- G15: Ermutigen zu granularem Feedback
- G16: System vermittelt die Konsequenzen von Benutzer:innenaktionen
- G17: Globale Kontrollen vorsehen

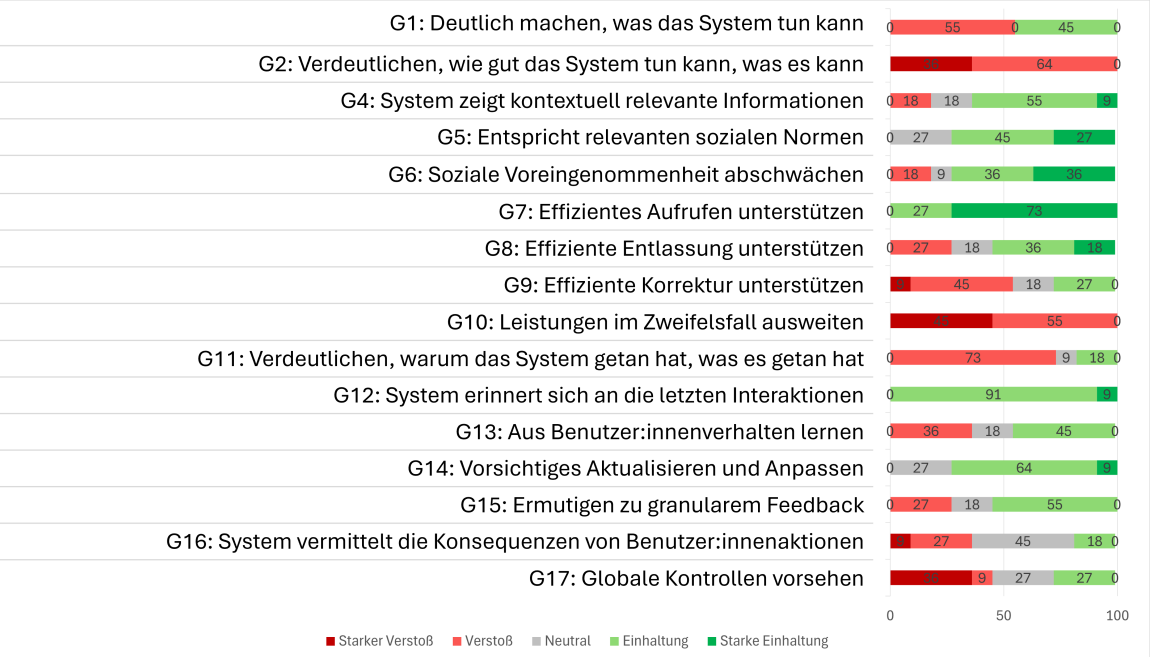


Abbildung 4: Von Teilnehmenden gemeldete Verstöße gegen die Richtlinien für Mensch-KI-Interaktion, nach [13]

Wie sehr oder wenig die Richtlinien laut den Befragten befolgt wurden, ist in Abbildung 4 dargestellt. Die Richtlinien, die insbesondere verletzt wurden, sind G1, G2, G9, G10 und G11. Choudhuri u. a. [13] gehen des Weiteren darauf ein, welche Folgen die Verletzungen dieser Richtlinien für das Arbeiten mit ChatGPT als Programmier-Assistenz haben. So ist ChatGPT nur eine begrenzte Hilfe bei der Arbeit mit Themen aus einer bestimmten Nische, außerdem hat es nicht immer die Ziele und Probleme der Teilnehmenden verstanden, was das Frustrationslevel der Teilnehmenden insbesondere erhöhte. Ein ähnliches Problem war, dass ChatGPT nur teilweise korrekte oder unvollständige Lösungen anbot. Darüber hinaus zeigten sich zwei weitere signifikante Probleme: Zum einen neigt ChatGPT zu Halluzinationen, bei denen es falsche Antworten generiert, wenn es die korrekte Lösung nicht kennt. Dies äußert sich in der Erfindung von nicht existierenden Funktionsparametern oder der Behauptung von Möglichkeiten, die tatsächlich nicht vorhanden sind. Zudem kann es zu einem Bestätigungsfehler kommen, bei dem ChatGPT dazu tendiert, den Aussagen oder Anfragen der Nutzer:innen zuzustimmen, unabhängig von deren tatsächlicher Richtigkeit oder Durchführbarkeit. Zum anderen gibt ChatGPT gelegentlich falsche Anleitungen. Dies tritt besonders dann auf, wenn das System das vorliegende Problem nicht vollständig

erfasst oder nur begrenztes Wissen zu einem bestimmten Thema hat. In solchen Fällen kann ChatGPT Fehler in den bereitgestellten Lösungen übersehen oder offensichtlich falsche oder unnötige Vorschläge unterbreiten.

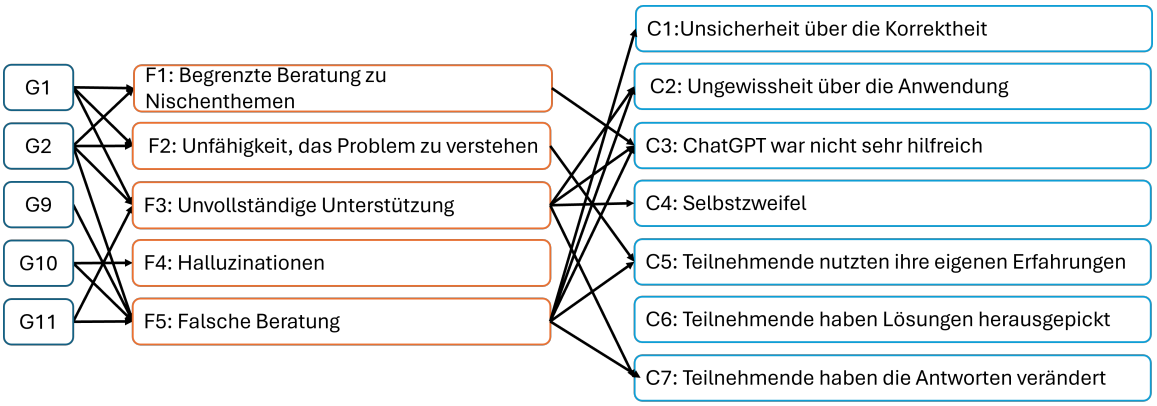


Abbildung 5: Zusammenhänge Verletzung der Richtlinien, fehlerhaftes Verhalten von ChatGPT und Reaktionen der Endnutzer:innen, nach [13]

Dies wiederum führt zu weiteren Folgen seitens der Teilnehmenden bzw. der Endnutzer:innen. Einige Teilnehmende beschrieben, dass sie unsicher waren, wie korrekt die Ausgaben von ChatGPT sind und wie sie angewendet werden sollten. ChatGPT wurde ebenfalls häufiger als nicht hilfreich angesehen, manche Teilnehmende zweifelten aber auch an sich selbst und suchten vermehrt Fehler bei sich selbst. Teilnehmende verzeichneten teilweise eher Erfolge, indem sie sich auf ihre eigene Erfahrung verließen, sich einzelne Lösungen oder Teile von Lösungen aus der Ausgabe von ChatGPT herauspickten oder die Ausgaben modifizierten.

Diese vorgestellten Abhängigkeiten sind in Abbildung 5 grafisch dargestellt, wobei die Pfeile je von Grund auf Ursache zeigen.

Choudhuri u. a. [13] zeigen mit ihren Ergebnissen, dass KI-Programmierassistenten zwar vielversprechend sind, aber noch erhebliche Herausforderungen in Bezug auf Benutzer:innenfreundlichkeit und Zuverlässigkeit bestehen. Entwickler:innen müssen oft den generierten Code anpassen oder verwerfen, was Fragen zur tatsächlichen Effizienzsteigerung durch diese Werkzeuge aufwirft. Insbesondere für Personen mit wenig Programmiererfahrung wie Student:innen oder Schüler:innen stellt die Nutzung von künstlicher Intelligenz nicht nur eine Chance dar, sondern auch eine Herausforderung.

Dies wird ebenfalls durch Erkenntnisse von Zviel-Girshin [67] untermauert. In ihrer Studie wurden 73 Teams mit je zwei Studierenden aus einem Programmier-Kurs für Erstsemester über einen Zeitraum von zwölf Wochen mit verschiedenen Aufgaben konfrontiert. Teilweise waren die Aufgaben gezielt für den Einsatz von künstlicher Intelligenz ausgelegt, die Studierenden erhielten aber keine Vorgaben, welche künstliche Intelligenz sie verwenden sollten. Zu beachten ist insbesondere, dass die Studierenden, bevor sie Aufgaben bearbeiten mussten, Beispiele gezeigt bekamen, wie man künstliche Intelligenzen als Werkzeuge beim Programmieren nutzen kann. Außerdem bekamen sie Tipps zur Formulierung effektiver Prompts. Am Anfang der Studie gaben lediglich 28 Prozent der Teams an, sich mit der Nutzung von KI-Werkzeugen vertraut zu fühlen. Am Ende des Kurses war diese

Zahl jedoch auf 100 Prozent gestiegen. Auch die Zufriedenheit der Studierenden mit den Antworten der jeweils gewählten künstlichen Intelligenz stieg im Laufe der Zeit deutlich an. Da die Studierenden jedoch während der Studie dasselbe Sprachmodell nutzten, deutet dieses Ergebnis darauf hin, dass sich ihre Fähigkeiten im Prompt-Engineering verbessert haben.

Zviel-Girshin [67] zeigt, dass mit Hilfestellung und Anleitung durch Lehrpersonal Grundlagen geschaffen werden können, die es Studierenden ermöglichen, gut mit künstlicher Intelligenz zu arbeiten und sie sinnvoll zur Lösung von Aufgaben einzusetzen. Gleichzeitig gibt es jedoch weiterhin die bereits von Liang, Yang und Myers [36] und Choudhuri u. a. [13] dargelegten Probleme der Unzuverlässigkeit sowie der teilweise schlechten Antwortqualität von KI-Werkzeugen. Eine Einführung in die Nutzung künstlicher Intelligenz durch Lehrpersonal kann Programmieranfänger:innen dabei helfen, schneller und mit höherer Zufriedenheit künstliche Intelligenz zu verwenden, aber die teilweise unzureichende Qualität der Antworten von künstlichen Intelligenzen bleibt ein zu lösendes Problem.

3.2 Erkennen von KI-generiertem Code

An dieser Stelle soll nach der kurzen Erklärung zur Erkennung von KI-generiertem Code aus dem Unterabschnitt 2.3 eine ausführlichere Vorstellung von Werkzeugen zur Erkennung von durch künstliche Intelligenz erzeugtem Code folgen. Hierzu wird die Studie von Pan u. a. [48] herangezogen, in der die Effektivität verschiedener KI-Detektoren bei der Unterscheidung zwischen menschlich geschriebenem und KI-generiertem Code untersucht wurde.

Die Forschenden entwickelten 13 Varianten von KI-generiertem Code, indem sie die Eingabeaufforderungen für ChatGPT oder den resultierenden Code modifizierten. Diese Varianten sind hier aufgeführt:

1. Ohne Modifikation: Verwendung unmodifizierter Codeprobleme mit ChatGPT, das repräsentiert den typischerweise gewählten Ansatz zur Problemlösung.

2. Stoppwort-Entfernung: Entfernung häufiger Stoppwörter[14] aus Codeproblemen, um Nutzer:innen-Modifikationen zur Verbesserung der Ausgabe zu simulieren.

3. Menschliche Nachahmung: Aufforderung an ChatGPT, menschenähnliche Antworten zu generieren, um kontextuell angemessenen Code zu erzeugen.

4. Lösung ohne Kommentare: ChatGPT anweisen, Kommentare auszulassen, um Fälle zu simulieren, in denen Programmierer:innen oder Schüler:innen und Studierende gute Programmierpraktiken verletzen.

5. Assertions-Testcode: ChatGPT wird aufgefordert, Assertions-Testcode einzuschließen, um die Fähigkeit von KI-Detektoren zur Erkennung gültigen Codes mit Testassertions zu bewerten.

6. Lösung mit Testfällen: ChatGPT soll hier Testfälle einschließen, um die Fähigkeit von KI-Detektoren zur Erkennung gültigen Codes mit Testfällen zu evaluieren.

[14] Z. B.: „der“, „die“, „das“, „in“, „an“, „von“, ...

7. Unittest-Testfälle: ChatGPT wird ähnlich wie in der Variante zuvor aufgefordert, Unittest-Testfälle einzuschließen, um die Fähigkeit von KI-Detektoren zur Erkennung gültigen Codes mit Unittest-Testfällen zu bewerten.

8. Variablennamen ersetzen: Alle Variablennamen im Code werden durch Einzelbuchstaben ersetzt, um Fälle zu simulieren, in denen grundlegende Namenskonventionen verletzt werden.

9. Funktionsnamen ersetzen: Alle Funktionsnamen durch Einzelbuchstaben ersetzt, um Fälle zu simulieren, in denen grundlegende Namenskonventionen verletzt werden.

10. Variablen- und Funktionsnamen ersetzen: Alle Variablen- und Funktionsnamen durch Einzelbuchstaben ersetzt, um extreme Verletzungen von Namenskonventionen zu simulieren.

11. Lange Methode: ChatGPT wird aufgefordert, längeren Code zu generieren, um Szenarien mit „Long Method Code Smell“ zu simulieren.

12. Kurze Methode: ChatGPT soll kürzeren Code generieren, um „Best Practices“ mit prägnanten, fokussierten Methoden zu simulieren.

13. „Toter“ Code: Es werden fünf Abschnitte mit „totem“ Code hinzugefügt, um den Einfluss auf die Leistung des KI-Detektors zu untersuchen und unbeabsichtigte Codeinklusion zu simulieren.

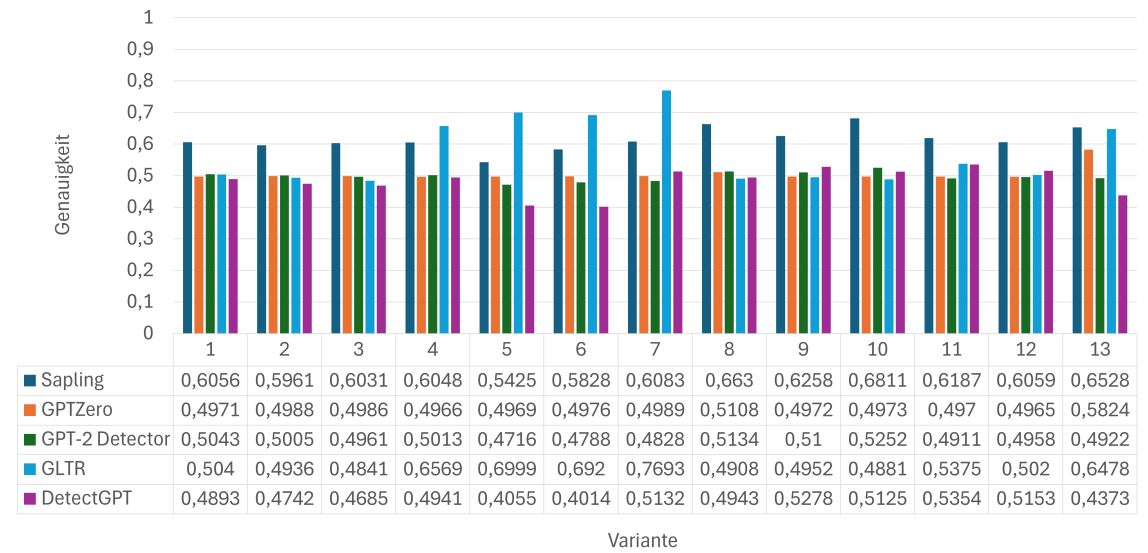
Die Untersuchung von fünf KI-Detektoren (*GPT-2 Detector*[15], *GPTZero*[16], *DetectGPT*[17], *GLTR*[18] und *Sapling*[19]) ergab, dass die meisten eine Genauigkeit von etwa 0,5 erreichten – also nicht besser als schlichtes Raten. Dabei wurden die „*True-Positive*“-Rate (menschengeschriebener Code wird als menschengeschrieben erkannt) und die „*True-Negative*“-Rate (KI-generierter Code wird als KI-generiert erkannt) betrachtet. Außerdem wurde die Genauigkeit der Detektoren für KI-generierten Code ermittelt. Sie berechnet sich folgendermaßen:

$$Genauigkeit = \frac{AnzahlTruePositive + AnzahlTrueNegative}{AnzahlCodes}$$

Dies ist eine wichtige Größe in Anbetracht der Anwendung der betrachteten Detektoren in Bildungseinrichtungen, um zu überprüfen, ob Schüler:innen und Studierende selbst geschriebenen Code im Unterricht oder in Prüfungssituationen verwenden – oder auch nicht.

Die Genauigkeiten der KI-Detektoren sind in Abbildung 6 zu sehen. In dieser Darstellung wird deutlich, dass sich nahezu alle Werte der Detektoren bei den verschiedenen Test-Varianten um 0,5 bewegen. Sie sind also kaum von Nutzen, um Code, der von Menschen geschrieben wurde, von KI-generiertem Code zu unterscheiden

[15]URL: `https://openai-openai-detector.hf.space/`, zuletzt aufgerufen: 20.12.2024
[16]URL: `https://gptzero.me/`, zuletzt aufgerufen: 20.12.2024
[17]URL: `https://detectgpt.com/?via=IAdvisor`
[18]URL: `http://gltr.io/`, zuletzt aufgerufen: 20.12.2024
[19]URL: `https://sapling.ai/ai-content-detector`, zuletzt aufgerufen: 20.12.2024



	1	2	3	4	5	6	7	8	9	10	11	12	13
Sapling	0,6056	0,5961	0,6031	0,6048	0,5425	0,5828	0,6083	0,663	0,6258	0,6811	0,6187	0,6059	0,6528
GPTZero	0,4971	0,4988	0,4986	0,4966	0,4969	0,4976	0,4989	0,5108	0,4972	0,4973	0,497	0,4965	0,5824
GPT-2 Detector	0,5043	0,5005	0,4961	0,5013	0,4716	0,4788	0,4828	0,5134	0,51	0,5252	0,4911	0,4958	0,4922
GLTR	0,504	0,4936	0,4841	0,6569	0,6999	0,692	0,7693	0,4908	0,4952	0,4881	0,5375	0,502	0,6478
DetectGPT	0,4893	0,4742	0,4685	0,4941	0,4055	0,4014	0,5132	0,4943	0,5278	0,5125	0,5354	0,5153	0,4373

Abbildung 6: Genauigkeit der Detektoren für KI-generierten Code, nach [48, 4]

GPT-2 Detector und GPTZero tendierten allgemein dazu, einen Großteil des Codes fälschlicherweise als von einem Menschen geschrieben zu klassifizieren [48]. Sie erreichen durchschnittlich eine Genauigkeit von etwa 0,4972 und 0,5050. DetectGPT zeigte bei einigen Varianten eine leichte Verbesserung, blieb aber insgesamt unzuverlässig bei einer durchschnittlichen Genauigkeit von etwa 0,4822. GLTR wies die größte Leistungsvariabilität auf, mit Genauigkeiten zwischen 0,48 und 0,77 je nach Variante und einer durchschnittlichen Genauigkeit von ca. 0,5739. Sapling schnitt am besten ab, erreichte aber bei den meisten Varianten nur Genauigkeiten knapp über 0,6 bei einer durchschnittlichen Genauigkeit von etwa 0,6147.

Die Studie identifizierte weiterhin mehrere Limitationen der Detektoren. Zum einen sind die meisten für natürliche Sprache optimiert und haben Schwierigkeiten mit der Struktur und Syntax des Programmcodes. Zum anderen sind sie oft auf bestimmte Modelle künstlicher Intelligenz trainiert und können Code von anderen Modellen schlechter erkennen. Zu guter Letzt ist noch zu beachten, dass es insgesamt und auch für Menschen nicht leicht ist, KI-generierten Code zu erkennen, da er von der Struktur her vielen Regeln folgt und deutlich weniger Varianten aufweisen kann als ein ähnlich langer Text, der keinen Regeln unterworfen ist.

Problematisch ist, dass bestimmte Varianten (insbesondere die Varianten 2, 3 und 5) es Schüler:innen ermöglichen könnten, die meisten Detektoren zu täuschen. In Anbetracht der Möglichkeiten, die sich Schüler:innen durch die hohe Verfügbarkeit künstlicher Intelligenz zum unehrlichen Ablegen von Programmier-Prüfungen bieten, sollten sich Lehrkräfte nicht auf diese Detektoren verlassen. Dabei ist nicht nur zu bedenken, dass Schüler:innen mit Betrugsversuchen durchkommen könnten, sondern auch, dass ehrliche Schüler:innen fälschlicherweise angeklagt werden zu betrügen und ihnen im schlimmsten Fall Leistungen nicht anerkannt werden.

Pan u. a. [48] betonen, dass die Ergebnisse die Notwendigkeit weiterer Forschung und Entwicklung in diesem Bereich unterstreichen. Es besteht ein dringender Bedarf

an effektiveren Methoden zur Unterscheidung zwischen menschlich geschriebenem und KI-generiertem Code, insbesondere im Kontext der akademischen Integrität und der Bewertung von Programmieraufgaben in Bildungseinrichtungen. Lehrkräfte können sich nicht auf aktuelle Detektoren von KI-generiertem Code verlassen, da diese kaum besser als zufälliges Raten sind. Die Wahrscheinlichkeit, einen von einer künstlichen Intelligenz geschriebenen Code fälschlicherweise als menschengeschrieben anzuerkennen oder einen menschengeschriebenen Code als KI-generiert zu klassifizieren, ist noch zu hoch, als dass man die vorgestellten Detektoren als alleinige Grundlage der Bewertung einsetzen könnte.

Zukünftige Forschung sollte sich auf die Entwicklung spezialisierter Detektoren konzentrieren, die die speziellen Eigenschaften von Programmcode berücksichtigen und robuster gegenüber verschiedenen Manipulationstechniken sind.

4 Herausforderungen und Chancen

Künstliche Intelligenz hat das Potenzial, die Bildungslandschaft erheblich zu verändern, sowohl in der allgemeinen Bildung als auch konkret in der Programmierausbildung. Während die Integration von künstlicher Intelligenz viele Chancen bietet, sind auch Herausforderungen und Risiken zu beachten. Im Folgenden sollen verschiedene dieser Aspekte detaillierter erörtert werden.

Allgemein kann man sagen, dass künstliche Intelligenz als eine Art Starthilfe fungieren kann, beispielsweise im Schreibprozess und beim Programmieren [6, 29]. Sie ermöglicht es Lernenden, schnell auf Informationen zuzugreifen oder stellt ihnen einen Rahmen zur Verfügung, den sie mit eigenen Ideen füllen können [6]. So unterstützt künstliche Intelligenz kreative Prozesse, indem sie Vorschläge und Lösungen für verschiedenste Probleme bereitstellt [50]. Dies kann insbesondere für Anfänger:innen hilfreich sein, die sich in neuen Themenbereichen zurechtfinden müssen.

4.1 Vor- und Nachteile von KI in der Bildung

Im Kontext der Bildung von Kindern oder Erwachsenen ermöglicht künstliche Intelligenz eine sofortige Beantwortung von Fragen und kann dazu beitragen, das Lernen zu einem leichteren und angenehmeren Erlebnis zu machen. Schüler:innen brauchen sich zudem für vermeintlich „dumme“ Fragen nicht zu schämen und sind im privaten Gespräch mit einer künstlichen Intelligenz eher bereit, Unwissenheit zuzugeben [39]. Adaptive Lernsysteme mit künstlicher Intelligenz können den Fortschritt und die individuellen Bedürfnisse einzelner Schüler:innen analysieren und ihren Unterricht durch individuelle Lernpläne entsprechend anpassen [17, 50].

Auch für Lehrkräfte kann künstliche Intelligenz Vorteile bieten, etwa indem sie bei der Bewertung und Korrektur von Aufgaben unterstützt oder den Unterricht plant [17]. Außerdem sind künstliche Intelligenzen hilfreich, um festzustellen, ob Schüler:innen ihre Aufgaben mit künstlicher Intelligenz bearbeitet haben bzw. in unerlaubter Weise mit künstlicher Intelligenz bearbeitet haben [6].

Denn auch die negative Seite bezüglich künstlicher Intelligenz im Bildungssektor muss betrachtet werden. Künstliche Intelligenz bietet enormes Potenzial, um von Schüler:innen

genutzt zu werden, um bei Aufgaben zu schummeln oder ganze Prüfungen automatisch ablegen zu lassen [6]. Dabei haben es die unehrlichen Schüler:innen wesentlich leichter als in den Zeiten vor künstlicher Intelligenz, als man die Antworten im Internet recherchieren musste oder sich mit anderen ausgetauscht hat. Heute reicht eine Frage an die künstliche Intelligenz, und man bekommt eine perfekt auf die jeweilige Frage zugeschnittene Antwort, zudem ist diese jederzeit verfügbar [6]. Um dem entgegenzuwirken oder zumindest die schummelnden Schüler:innen zu entlarven, bietet es sich an, Prüfungen vor Ort mit eingeschränkten technischen Hilfsmitteln oder ganz ohne technische Hilfsmittel durchzuführen.

Ein weiterer Kritikpunkt am Einsatz künstlicher Intelligenz in der Bildung ist das Risiko, dass sich Schüler:innen zu sehr auf die Ergebnisse der künstlichen Intelligenz verlassen und dadurch nicht mehr selbst über Probleme nachdenken und ihre Fähigkeit zum kreativen Denken und Aufgaben lösen darunter leidet [48, 1]. Dass sich Schüler:innen blind auf Ergebnisse künstlicher Intelligenz verlassen, ist umso gravierender in Anbetracht der Tatsache, dass die Quellen, aus denen die künstliche Intelligenz ihre Informationen bezieht, unbekannt, ungenau oder schlichtweg falsch sein können [14]. Unter diesem Aspekt ist es umso wichtiger, dass Schüler:innen beigebracht wird, kritisch zu denken und Informationen selbst nachzuprüfen – insbesondere im Umgang mit künstlicher Intelligenz [1].

4.2 Vor- und Nachteile von KI in der Programmierausbildung

Was für die Chancen und Risiken bei dem Umgang mit künstlicher Intelligenz in Bildung allgemein gilt, gilt insbesondere auch für den Einsatz künstlicher Intelligenz in der Programmierausbildung. Die Integration von künstlicher Intelligenz in die Programmierausbildung bietet weitere spezifische Vor- und Nachteile, die über die allgemeinen Auswirkungen auf die Bildung hinausgehen. Diese zusätzlichen Aspekte verdienen besondere Aufmerksamkeit, da künstliche Intelligenz die Art und Weise, wie Programmieren in Zukunft gelehrt und gelernt wird, grundlegend verändern können.

Ein konkreter Nutzen zeigt sich beim Erlernen von objektorientierten Programmiersprachen [14]. KI-Systeme können komplexe Konzepte veranschaulichen und praktische Beispiele liefern, die das Verständnis der Schüler:innen fördern. Dies ist besonders wertvoll, da abstrakte Objekte wie Klassen der objektorientierten Programmierung für Programmier-Anfänger:innen manchmal schwer zu verstehen sind [49]. Ein weiterer signifikanter Vorteil ist die Unterstützung beim Verstehen von Fehlermeldungen. Künstliche Intelligenz kann teils kryptische Compiler-Meldungen in verständliche Erklärungen übersetzen, den Programmierer:innen erklären, wo das Problem liegt, und Lösungsvorschläge anbieten [6, 52]. Das kann die Frustration reduzieren und den Lernprozess beschleunigen. Aber auch, wenn keine Fehlermeldung vorliegt, sondern stattdessen ein semantischer Fehler, kann künstliche Intelligenz hilfreich sein, indem sie den Code überprüft und nötige Änderungen vorschlägt. KI-Werkzeuge können auch bei der Automatisierung einfacher, aber zeitaufwändiger Aufgaben helfen, wie dem Schreiben von Getter- und Setter-Methoden oder der Erstellung von Dokumentation [36]. Dies ermöglicht es den Schüler:innen, sich auf komplexere Aspekte des Programmierens zu konzentrieren und ihnen mehr Zeit zu widmen. Besonders wertvoll ist die Unterstützung künstlicher Intelligenz auch bei der

Strukturierung und Planung größerer Projekte. Künstliche Intelligenz kann Vorschläge zur Architektur und zum Design liefern, was Programmier-Anfänger:innen helfen kann, gute Praktiken zu erlernen und anzuwenden [16]. Schließlich kann künstliche Intelligenz wie zuvor schon angesprochen als Ideengenerator dienen, indem sie eine erste Codebasis liefert, auf die Schüler:innen aufbauen können. Dies kann den kreativen Prozess anregen und dabei helfen, Startblockaden zu überwinden [6].

Die grundlegende Frage bleibt jedoch, wie umfassend künstliche Intelligenz tatsächlich beim Lernen des Programmierens hilfreich sein kann. Während künstliche Intelligenz bei spezifischen Aufgaben und der Vermittlung von Wissen hilfreich sein kann, ist unklar, ob sie die Entwicklung von Problemlösungsfähigkeiten und algorithmischem Denken in gleichem Maße fördern kann. Es besteht die Gefahr, dass Schüler:innen nur oberflächliches Wissen entwickeln, ohne die zugrunde liegenden Prinzipien zu verstehen [13].

Eine zentrale Sorge ist ebenso, dass sich die Schüler:innen zu sehr an die ständige Hilfe der künstlichen Intelligenz gewöhnen und möglicherweise die grundlegende Syntax der Programmiersprachen nicht mehr erlernen [6]. Dies wirft jedoch die Frage auf, ob das detaillierte Erlernen von Syntax in einer Welt, in der KI-Assistenten allgegenwärtig sind, überhaupt noch notwendig ist [6]. KI-Systeme sind, auch wenn sie beim Programmieren unterstützen sollen, nicht perfekt und können Fehler machen oder irreführende Informationen liefern. Dies kann zu Frustration bei den Lernenden führen, insbesondere wenn sie noch nicht über die Erfahrung verfügen, KI-generierte Vorschläge kritisch zu hinterfragen [14].

Selbst wenn die künstliche Intelligenz korrekten Code liefert, ist es möglich, dass der Code, auf dem das KI-Modell trainiert wurde und den es jetzt (in abgewandelter Form) wiedergibt, zu komplex für Anfänger:innen ist [6, 32]. Es kann zu Verwirrung und Frustration führen, wenn die künstliche Intelligenz Lösungen vorschlägt, die deutlich über dem Niveau der Schüler:innen liegen [32]. Allerdings bietet sich hier die Möglichkeit, die künstliche Intelligenz selbst um Erklärungen zu dem Code zu bitten, was wiederum als Lernchance für die Schüler:innen genutzt werden kann.

4.3 Allgemeine Bedenken

Neben den Risiken bezüglich des Einsatzes von künstlicher Intelligenz in der Bildung oder der Programmierausbildung gibt es noch weitere Nachteile, die nicht außer Acht gelassen werden sollten.

Beispielsweise gibt es rechtliche Risiken, insbesondere im Hinblick auf Plagiarismus. Die Nutzung von künstlicher Intelligenz zur Erstellung von Inhalten kann dazu führen, dass die Grenzen zwischen eigener Arbeit und KI-generierten Inhalten, die auf Inhalten Dritter basieren, verschwimmen [36]. Es stellt sich die Frage bezüglich der Urheberschaft: Ab wann ist ein Text oder der Code eines Programmes nicht mehr das Werk der Person, die die künstliche Intelligenz um Hilfe gefragt hat? Wessen Werk ist es stattdessen, das der künstlichen Intelligenz oder das der Personen, die die Werke erschaffen haben, die die künstliche Intelligenz als Trainingsdaten genutzt hat? [6, 36]

Auch der Energieverbrauch für das Training und die Benutzung einer künstlichen Intelligenz ist nicht unerheblich und sollte dazu anhalten, künstliche Intelligenz nur gezielt

und bei passenden Aufgaben einzusetzen [6]. Eine Anfrage an eine künstliche Intelligenz verbraucht etwa 60-mal mehr Energie als eine Suche mit einer Suchmaschine [58].

Ein weiteres Risiko ist die Verstärkung von Vorurteilen. Wenn KI-Systeme auf voreingenommenen Daten trainiert werden, können sie bestehende Ungleichheiten reproduzieren oder sogar verstärken [6].

Die Nutzung von künstlicher Intelligenz im Beruf wirft ebenfalls Bedenken auf. Während künstliche Intelligenz die Produktivität steigern kann, besteht die Gefahr, dass sich auch erfahrene Mitarbeitende zu sehr auf automatisierte Systeme verlassen und dadurch ihre eigenen Fähigkeiten vernachlässigen oder Fehler übersehen [36]. Sicherheitskritische Informationen dürfen nicht in KI-Systeme eingegeben werden, was eine Herausforderung für deren Anwendung in sensiblen Bereichen darstellt. Dem kann durch Schulungen der Mitarbeitenden entgegengewirkt werden oder auch durch eine unternehmensinterne künstliche Intelligenz, die sensible Daten verarbeiten kann, aber nicht nach außen weitergibt.

Die Integration von künstlicher Intelligenz in Bildung und Programmierung bietet zahlreiche Chancen zur Verbesserung des Lernens und zur Steigerung der Produktivität. Gleichzeitig müssen jedoch auch die damit verbundenen Risiken sorgfältig betrachtet werden. Eine ausgewogene Herangehensweise ist notwendig, um sicherzustellen, dass die Vorteile maximiert und die Herausforderungen minimiert werden. Im folgenden Kapitel werden Empfehlungen diskutiert, die dieses Ziel erreichen sollen.

5 Empfehlungen

Die zunehmende Verbreitung von künstlicher Intelligenz in der Programmierausbildung stellt Bildungseinrichtungen vor neue Herausforderungen. Da sich die Nutzung solcher Werkzeuge durch Schüler:innen und Studierende kaum verhindern lässt, ist es ratsam, einen proaktiven Ansatz zu wählen und den korrekten Umgang damit zu lehren. Gleichzeitig müssen Methoden entwickelt werden, wie mit künstlicher Intelligenz in Prüfungssituationen umgegangen werden kann. Parallel dazu sollten sowohl KI-Systeme als auch KI-Detektoren kontinuierlich verbessert werden – insbesondere bei der Generierung von Code und der Detektion von diesem KI-generierten Code.

Um künstliche Intelligenz effektiv in die Bildung zu integrieren, sollten Lehrende zunächst klare Lernziele definieren, die mit dem Einsatz von künstlicher Intelligenz vereinbar sind. Es muss sorgfältig abgewogen werden, welcher Grad der Automatisierung angemessen ist – von vollständiger Automatisierung bis hin zu einem ergänzenden Ansatz, der KI-Fähigkeiten mit menschlicher Beteiligung kombiniert [48].

Es ist wichtig, künstliche Intelligenz als effektive Lernhilfe zu vermitteln. Direkte Lösungsvorschläge können kritisches Denken beeinträchtigen und die Lernmotivation senken. Stattdessen sollten KI-Systeme entwickelt werden, die die Absichten der Lernenden korrekt interpretieren und ihre Interaktionen an pädagogische Ziele anpassen können [13].

Dabei darf man aber nicht außer Acht lassen, dass auch den Schüler:innen deutlich gemacht werden muss, dass künstliche Intelligenz hinterfragt werden muss.

Ethische Aspekte spielen ebenfalls eine zentrale Rolle. Es müssen umfassende Richtlinien und Strategien entwickelt werden, um einen verantwortungsvollen und ethischen Einsatz von künstlicher Intelligenz im Bildungskontext zu gewährleisten. Dazu gehört auch eine kontinuierliche Evaluation der Effektivität von künstlicher Intelligenz in der Bildung [48].

Für Lehrende im Bereich der objektorientierten Programmierung ist es ratsam, bei der Bewertung mehr Gewicht auf Codequalität, Designmuster und ähnliche Aspekte zu legen. Der Fokus sollte sich von der reinen Funktionalität hin zu „funktionalem und hochwertigem Code“ [14] verschieben. Zudem sollten KI-Modelle aktiv in den Unterricht einbezogen werden, etwa durch Übungen, bei denen Studierende mit künstlicher Intelligenz interagieren, um Code zu generieren und anschließend zu evaluieren [14].

Die Integration von KI-generiertem Code in die Programmierausbildung erfordert eine Anpassung der pädagogischen Ansätze in Schulen und anderen Bildungseinrichtungen. So könnte man den Schwerpunkt vom reinen Programmieren hin zum Lesen und Bewerten von Code verschieben. Zudem sollten ethische Überlegungen von Beginn an in den Lehrplan integriert werden, um Studierende für dieses Thema zu sensibilisieren [6].

Projektbasiertes Lernen kann eine effektive Methode sein, um komplexere Interaktionen mit KI-Modellen zu fördern und gleichzeitig die Anwendung objektorientierter„Best Practices“ zu üben [14].

Abschließend ist es wichtig zu betonen, dass die Bildungslandschaft sich schneller denn je verändert – ohne schnelle und konzertierte Bemühungen riskieren Lehrkräfte, die Chance zu verpassen, die Entwicklung und Nutzung von künstlicher Intelligenz in der Bildung aktiv mitzugestalten. Es liegt insbesondere in der Verantwortung der Bildungseinrichtungen, Schüler:innen und auch Studierende auf eine Zukunft vorzubereiten, in der KI-unterstützte Softwareentwicklung zur Norm werden wird [6].

6 Zusammenfassung und Ausblick

Die Untersuchung zeigt, dass künstliche Intelligenz in der Bildung und insbesondere in der Programmierausbildung zukünftig sowohl Chancen als auch Herausforderungen bereithält. KI-gestützte Tools revolutionieren die Art und Weise, wie Programmcode geschrieben und das Programmieren gelernt wird. Diese Technologien bieten enorme Vorteile in Bezug auf Produktivität, Fehlerreduzierung und Unterstützung beim Erlernen neuer Programmiersprachen. Gleichzeitig stellen diese Entwicklungen die traditionelle Programmierausbildung vor neue Herausforderungen. Es wird zunehmend wichtiger, Schüler:innen und Studierende nicht nur in der Syntax und Logik von Programmiersprachen zu schulen, sondern auch in der effektiven Nutzung und kritischen Bewertung von KI-generierten Codevorschlägen. Die Fähigkeit, mit KI-Systemen zu interagieren, kann zu einer Kernkompetenz für zukünftige Softwareentwickler:innen werden.

Die Integration von künstlicher Intelligenz in die Programmierausbildung erfordert eine sorgfältige Balance. Einerseits müssen die Grundlagen der Programmierung weiterhin vermittelt werden, um ein tiefes Verständnis der zugrunde liegenden Konzepte zu gewährleisten. Insbesondere muss die korrekte Anwendung dieser Kenntnisse bei den Schüler:innen und Studierenden überprüfbar sein, allerdings gibt es bisher keine ausreichend guten Detektoren für KI-generierten Code. Andererseits muss die Ausbildung die

Schüler:innen und Studierenden auf eine Arbeitswelt vorbereiten, in der KI-Assistenten ein integraler Bestandteil des Entwicklungsprozesses sind.

Für die Zukunft ist es entscheidend, dass Bildungseinrichtungen proaktiv Strategien entwickeln, um künstliche Intelligenz sinnvoll in den Unterricht zu integrieren. Dies beinhaltet die Anpassung von Lehrplänen, die Schulung von Lehrkräften und die Entwicklung ethischer Richtlinien für den Einsatz von künstlicher Intelligenz in Prüfungssituationen.

Es wird weitere Forschung notwendig sein, um die langfristigen Auswirkungen von künstlicher Intelligenz auf die Programmierkompetenzen von Schüler:innen und Studierenden zu untersuchen und effektive pädagogische Konzepte für das KI-unterstützte Lernen zu entwickeln. Zudem sollten die rechtlichen und ethischen Rahmenbedingungen für den Einsatz von künstlicher Intelligenz in der Bildung kontinuierlich überprüft und angepasst werden.

Insgesamt zeigt sich, dass künstliche Intelligenz das Potenzial hat, die Programmierausbildung grundlegend zu verändern. Es liegt an Bildungseinrichtungen, Lehrenden und politischen Entscheidungsträger:innen, diesen Wandel so zu gestalten, dass er den Bedürfnissen der Lernenden und den Anforderungen der sich schnell entwickelnden digitalen Welt gerecht wird.

Literatur

[1] Md Mokshud Ali u. a. „Gen Z and Generative AI: Shaping the Future of Learning and Creativity“. In: (2024).

[2] Saleema Amershi u. a. „Guidelines for Human-AI Interaction“. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. Glasgow, Scotland Uk: Association for Computing Machinery, 2019, S. 1–13. ISBN: 9781450359702. DOI: 10.1145/3290605.3300233. URL: https://doi.org/10.1145/3290605.3300233.

[3] Anjuman Ara, Md Sajadul Alam u. a. „A comparative review of AI-generated image detection across social media platforms“. In: *Global Mainstream Journal of Innovation, Engineering & Emerging Technology* 3.01 (2024), S. 11–22.

[4] Anonymous Author. „Variant Result“. In: (Okt. 2023). DOI: 10.6084/m9.figshare.24265015.v1. URL: https://figshare.com/articles/dataset/Variant_Result/24265015.

[5] Ömer Aydın. „Google Bard generated literature review: metaverse“. In: *Journal of AI* 7.1 (2023), S. 1–14.

[6] Brett A. Becker u. a. „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, Volume 1, SIGCSE 2023, Toronto, ON, Canada, March 15-18, 2023*. Hrsg. von Maureen Doyle u. a. ACM, 2023, S. 500–506. DOI: 10.1145/3545945.3569759. URL: https://doi.org/10.1145/3545945.3569759.

[7] Valentina Bellini u. a. „Between human and AI: assessing the reliability of AI text detection tools“. In: *Current Medical Research and Opinion* 40.3 (2024), S. 353–358.

[8] Kaiyi Cao. „2023 International Conference on Digital Economy and Business Administration (ICDEBA 2023)“. In: *SHS Web of Conferences* 181 (2024), S. 6. DOI: 10.1051/shsconf/202418101009.

[9] Guisseppi Paul Morales Cauti u. a. „Intelligent Video Surveillance: Artificial Intelligence and its Applications on Security Systems“. In: *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*. IEEE. 2023, S. 986–991.

[10] Induri Chandana u. a. „Detecting AI Generated Text“. In: *2024 2nd World Conference on Communication & Computing (WCONF)*. 2024, S. 1–6. DOI: 10.1109/WCONF61366.2024.10692028.

[11] Subhash Chander. „IMPACT OF ARTIFICIAL INTELLIGENCE ON SOCIETY: RISK AND CHALLENGES“. In: *International Journal of Engineering Science and Humanities* 14.Special Issue 1 (2024), S. 103–111.

[12] Lijia Chen, Pingping Chen und Zhijian Lin. „Artificial Intelligence in Education: A Review“. In: *IEEE Access* 8 (2020), S. 75264–75278. DOI: 10.1109/ACCESS.2020.2988510.

[13] Rudrajit Choudhuri u. a. „How Far Are We? The Triumphs and Trials of Generative AI in Learning Software Engineering“. In: *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 2024, S. 1–13.

[14] Bruno Pereira Cipriano und Pedro Alves. „LLMs Still Can't Avoid Instanceof: An Investigation Into GPT-3.5, GPT-4 and Bard's Capacity to Handle Object-Oriented Programming Assignments“. In: *Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training*. 2024, S. 162–169.

[15] Eryka Clarencia u. a. „The Impact of Artificial Intelligence in the Creative Industries: Design and Editing“. In: *2024 International Seminar on Application for Technology of Information and Communication (iSemantic)*. IEEE. 2024, S. 440–444.

[16] B. Drabble. „Artificial intelligence for project planning“. In: *IEE Colloquium on Future Developments in Projects Management Systems*. 1995, S. 3/1–3/5. DOI: 10.1049/ic:19951125.

[17] Tong Feng und Qinglun Li. „Artificial Intelligence in Education Management: Opportunities, Challenges, and Solutions“. In: *Frontiers in Business, Economics and Management* 16.3 (2024). Corresponding author: Tong Feng (Email: 596928182@qq.com), S. 449. ISSN: 2766-824X. DOI: 10.54097/raxsbp45.

[18] Aleksandar Filipović. „The Role of Artificial Intelligence in Video Game Development“. In: *Kultura Polisa* 20.3 (2023), S. 50–67.

[19] James Finnie-Ansley u. a. „My AI Wants to Know if This Will Be on the Exam: Testing OpenAI's Codex on CS2 Programming Exercises“. In: *Proceedings of the 25th Australasian Computing Education Conference*. ACE '23. Melbourne, VIC, Australia: Association for Computing Machinery, 2023, S. 97–104. ISBN: 9781450399418. DOI: 10.1145/3576123.3576134. URL: https://doi.org/10.1145/3576123.3576134.

[20] Manasi Gaikwad und Ayesha Khan. „The Role of Artificial Intelligence in Advancing Climate Change Research“. In: *International Journal for Research in Applied Science & Engineering Technology (IJRASET)* 12.VI (Juni 2024). IC Value: 45.98, SJ Impact Factor: 7.538, ISRA Journal Impact Factor: 7.894. ISSN: 2321-9653. URL: https://www.ijraset.com.

[21] Lalit Gupta. „Unmasking artificial intelligence (AI): Identifying articles written by AI models“. In: *Indian Journal of Clinical Anaesthesia* (2024).

[22] Chris Huntingford u. a. „Machine learning and artificial intelligence to aid climate change research and preparedness“. In: *Environmental Research Letters* 14.12 (2019), S. 124007.

[23] Tariqul Islam u. a. „Artificial Intelligence in Fraud Detection and Financial Risk Mitigation: Future Directions and Business Applications“. In: *IJFMR* (2024).

[24] Dhiraj Jadhav u. a. „AI-Driven Text-to-Multimedia Content Generation: Enhancing Modern Content Creation“. In: *2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE. 2024, S. 1610–1615.

[25] Tarun Jagadish und S Graceline Jasmine. „Detection of AI-Generated Image Content in News and Journalism“. In: *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. 2024, S. 1–6. DOI: 10.1109/ICCCNT61001.2024.10724589.

[26] Y. Jia. „A Comprehensive Review of Diffusion Models in AI-Generated Content for Image Applications“. In: *Applied and Computational Engineering* 94 (2024), S. 197–202.

[27] Zhengyuan Jiang u. a. „Watermark-based Attribution of AI-Generated Content“. In: 2024. URL: https://api.semanticscholar.org/CorpusID:268987647.

[28] Zixuan Jiang. „Emotional Simulation in Game AI and Its Impact on Player Experience“. In: *International Journal of Education and Humanities* 13.2 (2024), S. 11–13.

[29] Martin Jonsson und Jakob Tholander. „Cracking the code: Co-coding with AI in creative programming education“. In: *C&C '22: Creativity and Cognition, Venice, Italy, June 20 - 23, 2022*. ACM, 2022, S. 5–14. DOI: 10.1145/3527927.3532801. URL: https://doi.org/10.1145/3527927.3532801.

[30] Mehar Prateek Kalra, Ansh Mathur und C Patvardhan. „Detection of AI-generated Text: An Experimental Study“. In: *2024 IEEE 3rd World Conference on Applied Intelligence and Computing (AIC)*. 2024, S. 552–557. DOI: 10.1109/AIC61668.2024.10731116.

[31] H Kamel. „Artificial intelligence for predictive maintenance“. In: *Journal of Physics: Conference Series*. Bd. 2299. 1. IOP Publishing. 2022, S. 012001.

[32] Majeed Kazemitabaar u. a. „How Novices Use LLM-based Code Generators to Solve CS1 Coding Tasks in a Self-Paced Learning Environment“. In: *Proceedings of the 23rd Koli Calling International Conference on Computing Education Research*. Koli Calling '23. Koli, Finland: Association for Computing Machinery, 2024. ISBN: 9798400716539. DOI: 10.1145/3631802.3631806. URL: https://doi.org/10.1145/3631802.3631806.

[33] Ralf T Kreutzer u. a. „Fields of application of artificial intelligence—Security sector and military sector“. In: *Understanding Artificial Intelligence: Fundamentals, Use Cases and Methods for a Corporate AI Journey* (2020), S. 225–233.

[34] Bing Li u. a. „Advances and challenges in artificial intelligence text generation“. In: *Frontiers of Information Technology & Electronic Engineering* 25.1 (2024), S. 64–83.

[35] Yujia Li u. a. „Competition-level code generation with AlphaCode“. In: *Science* 378.6624 (2022), S. 1092–1097. DOI: 10.1126/science.abq1158. eprint: https://www.science.org/doi/pdf/10.1126/science.abq1158. URL: https://www.science.org/doi/abs/10.1126/science.abq1158.

[36] Jenny T Liang, Chenyang Yang und Brad A Myers. „A large-scale survey on the usability of ai programming assistants: Successes and challenges“. In: *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. 2024, S. 1–13.

[37] Maria Teresa Linaza u. a. „Data-driven artificial intelligence applications for sustainable precision agriculture“. In: *Agronomy* 11.6 (2021), S. 1227.

[38] Spyros Makridakis, Fotios Petropoulos und Yanfei Kang. „Large language models: Their success and impact“. In: *Forecasting* 5.3 (2023), S. 536–549.

[39] Eduina Maksuti und Isa Erbas. „The Impact of Artificial Intelligence on Education“. In: *Multidisciplinary Joint Akseprin Journal* 2.2 (2024), S. 11–20.

[40] Hazem W Marar. „Advancements in software engineering using AI“. In: *Computer Software and Media Applications* 6.1 (2024), S. 3906.

[41] K Mashood u. a. „ARTIFICIAL INTELLIGENCE RECENT TRENDS AND APPLICATIONS IN INDUSTRIES“. In: *Pakistan Journal of Science* 75.02 (2023).

[42] Shervin Minaee u. a. „Large language models: A survey“. In: *arXiv preprint arXiv:2402.06196* (2024).

[43] Minkorrekt. *Mi318 – „Nici – Allein zu Haus“*. https://minkorrekt.de/mi318-nici-allein-zu-haus/. 00:32:25: Interview mit Henri von ScienceOS, zuletzt aufgerufen am 23.01.2025. Okt. 2024.

[44] Radhika Nautiyal u. a. „Intersection of Artificial Intelligence (AI) in Entertainment Sector“. In: *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*. IEEE. 2023, S. 1273–1278.

[45] Bernhard Nebel. „Turing-Test“. In: *Mensch-Maschine-Interaktion: Handbuch zu Geschichte–Kultur–Ethik*. Springer, 2019, S. 304–306.

[46] Muyao Niu. „Application and existing problems of artificial intelligence technology in the agricultural field“. In: *Applied and Computational Engineering* 35 (2024), S. 32–40.

[47] Kateryna Osadcha, Justyna Szynkiewicz und Mohd Sameen Chishti. „Using Microsoft Copilot Chat in the Work of IT Educators: Pilot Study“. In: *Norsk IKT-konferanse for forskning og utdanning*. 4. 2024.

[48] Wei Hung Pan u. a. „Assessing AI Detectors in Identifying AI-Generated Code: Implications for Education“. In: *Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training, SEET@ICSE 2024, Lisbon, Portugal, April 14-20, 2024*. ACM, 2024, S. 1–11. DOI: 10.1145/3639474.3640068. URL: https://doi.org/10.1145/3639474.3640068.

[49] K. G. D. K. Perera, J. Wijayanayake und J. Prasadika. „Factors Affecting the Effectiveness of Generative Artificial Intelligence Apps on University Students' Programming Language Learning in Sri Lanka: A Systematic Literature Review“. In: *2024 4th International Conference on Advanced Research in Computing (ICARC)*. 2024, S. 276–281. DOI: 10.1109/ICARC61713.2024.10499744.

[50] Zhang Qian. „Applications, risks and countermeasures of artificial intelligence in education“. In: *2021 2nd International Conference on Artificial Intelligence and Education (ICAIE)*. IEEE. 2021, S. 89–92.

[51] Shashishekhar Ramagundam und Niharika Karne. „The New Frontier in Media: AI-Driven Content Creation for Ad-Supported TV using Generative Adversarial Network“. In: *2024 7th International Conference of Computer and Informatics Engineering (IC2IE)*. IEEE. 2024, S. 1–6.

[52] Abhiraj Singh Rathore, Adarsh Sharma und Massoud Massoudi. „Personalized Engineering Education Model Based on Artificial Intelligence for Learning Programming“. In: *2021 6th International Conference on Computing, Communication and Security (ICCCS)*. 2021, S. 1–10. DOI: 10.1109/ICCCS51487.2021.9776343.

[53] Niklas Reini. „The Impact of AI powered code completion in the software engineering field.“ In: (2022).

[54] Vibhor Sharma, Monika Goyal und Drishti Malik. „An intelligent behaviour shown by chatbot system“. In: *International Journal of New Technology and Research* 3.4 (2017), S. 263312.

[55] Yan Sun und Yunna Liu. „Application and Prospects of Artificial Intelligence in Intelligent Transportation Systems“. In: *2023 International Conference on Internet of Things, Robotics and Distributed Computing (ICIRDC)*. 2023, S. 447–453. DOI: 10.1109/ICIRDC62824.2023.00088.

[56] L Brannon Thomas u. a. „Artificial intelligence: review of current and future applications in medicine“. In: *Federal Practitioner* 38.11 (2021), S. 527.

[57] Priyan Vaithilingam, Tianyi Zhang und Elena L. Glassman. „Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models“. In: *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*. CHI EA '22. New Orleans, LA, USA: Association for Computing Machinery, 2022. ISBN: 9781450391566. DOI: 10.1145/3491101.3519665. URL: https://doi.org/10.1145/3491101.3519665.

[58] Wim Vanderbauwhede. „Estimating the Increase in Emissions caused by AI-augmented Search“. In: *arXiv preprint arXiv:2407.16894* (2024).

[59] Michel Wermelinger. „Using GitHub Copilot to Solve Simple Programming Problems“. In: *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. SIGCSE 2023. Toronto ON, Canada: Association for Computing Machinery, 2023, S. 172–178. ISBN: 9781450394314. DOI: 10.1145/3545945.3569830. URL: https://doi.org/10.1145/3545945.3569830.

[60] Michael Mncedisi Willie. „Identifying AI-Generated Research Papers: Methods and Considerations: Methods and Considerations“. In: *Golden Ratio of Data in Summary* 4.2 (2024), S. 736–739.

[61] Zhenyu Xu und Victor S. Sheng. „Detecting AI-Generated Code Assignments Using Perplexity of Large Language Models“. In: *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*. Hrsg. von Michael J. Wooldridge, Jennifer G. Dy und Sriraam Natarajan. AAAI Press, 2024, S. 23155–23162. DOI: 10.1609/AAAI.V38I21.30361. URL: https://doi.org/10.1609/aaai.v38i21.30361.

[62] Xinyi Yang. „The applications of artificial intelligence in personalized medicine“. In: *Proceedings of the 6th International Conference on Computing and Data Science*. Sheffield, United Kingdom: DOI: 10.54254/2755-2721/71/20241625, 2024.

[63] Fan Yao u. a. „Human vs. Generative AI in Content Creation Competition: Symbiosis or Conflict?“ In: *arXiv preprint arXiv:2402.15467* (2024).

[64] Burak Yetiştiren u. a. „Evaluating the code quality of ai-assisted code generation tools: An empirical study on github copilot, amazon codewhisperer, and chatgpt“. In: *arXiv preprint arXiv:2304.10778* (2023).

[65] Chenyu Yu. „Research on the Application of Artificial Intelligence Technology in the Field of Intelligent Transportation Systems“. In: *Highlights in Science, Engineering and Technology* 83 (2024), S. 409–415.

[66] S. Zheng und M. Han. „The impact of AI enablement on students' personalized learning and countermeasures—A dialectical approach to thinking“. In: *Journal of Infrastructure, Policy and Development* 8.14 (2024), S. 10274. DOI: 10.24294/jipd10274.

[67] Rina Zviel-Girshin. „The Good and Bad of AI Tools in Novice Programming Education“. In: *Education Sciences* 14.10 (2024), S. 1089.