Der Einfluss von AI-Pair-Programmers auf die Digital Humanities: Potentiale und Limitationen

Schiller-Stoff, Sebastian
sebastian.stoff@uni-graz.at
Universität Graz, Österreich
ORCID: 0000-0001-6941-113X

Münzer, Leona Elisabeth
leona.muenzer@uni-graz.at
Universität Graz, Österreich
ORCID: 0009-0002-7170-8340

Dittmann, Christina
christina.dittmann@uni-graz.at
Universität Graz, Österreich
ORCID: 0009-0000-7085-3154

Sagadin, Suzana
suzana.sagadin@uni-graz.at
Universität Graz, Österreich

Einleitung

Es steht außer Frage, dass die *"Digitale Evolution"* und deren Konsequenzen nicht mehr aufzuhalten sind (Coester und Pohlmann 2021). Auch die sich im Zuge der *Digitalen Transformation* der Geisteswissenschaften etablierten *Digital Humanities* (kurz: DH) stehen ununterbrochen im Wandel, nicht zuletzt durch den verstärkten Einsatz von *Künstlicher Intelligenz* (Vogeler und Hofeneder 2023).

Gab es lange Zeit nur einfache Tools zur code completion, sind es mittlerweile sogenannte *AI-Pair-Programmers*, d.h. KI-gestützte, integrierbare Systeme wie GitHub Copilot, das bei seiner Veröffentlichung auf das large language model *Codex* von *OpenAI* basierte (Rothlauf et al. 2021), die in jüngster Zeit Aufmerksamkeit erlangt haben (Zhang et al. 2023). Diese können in IDEs (integrated development environments) integriert werden und so bei diversen Aufgaben unterstützen. *Aber was bieten diese Codingassistenten an?*

Auch auf die Programmiertätigkeiten der DH haben diese Assistenten immer mehr Einfluss: Wir befinden uns durchaus in einem Prozess, der in Zukunft dazu führen könnte, dass derartige Tools zum festen Bestandteil unserer Arbeitsweisen werden. Unsere Disziplin befindet sich also *"Under Construction" - aber welche (Forschungs-)Aktivitäten der DH sind wirklich davon betroffen, und inwiefern könnten sich diese verändern? Welche Potenziale gibt es?*

Grundlegende Funktionen von AI-Pair-Programmers

Neben *GitHub Copilot* sind derzeit verschiedene kommerzielle und nichtkommerzielle AI-Pair-Programmers, darunter auch Open-Source-Optionen, verfügbar. Am häufigsten werden diese als Plugins in *Microsofts VS Code* verwendet, jedoch sind diese zunehmend auch in andere IDEs wie den *JetBrains IDEs*, in diesem Fall seit Februar/März 2024, integrierbar (siehe dazu Schiller-Stoff und Münzer 2024) - letztere bieten mittlerweile ihre eigene Infrastruktur *JetBrains AI* an. Die jeweilige Leistung und der Umfang der Möglichkeiten dieser Assistenten ist natürlich von den zugrundeliegenden Daten abhängig, sie bieten aber ähnliche grundlegende Funktionen an (siehe dazu Martin und Schiller-Stoff 2023; Schiller-Stoff und Martin 2024):

Eine der Kernfunktionen ist die automatische Vervollständigung von Code: Mittels *code completion* kann Code automatisch vervollständigt werden, indem auf eigens trainierte Modelle für die Codegenerierung zurückgegriffen wird. Die Vervollständigung liefert kontextbezogene Vorschläge für Codezeilen, Funktionen und ganze Klassen, generiert Boilerplate-Code, der in vielen Projekten wiederkehrende Strukturen wie Initialisierungen, Standardmethoden oder Konfigurationsdateien umfasst. Auch ist es möglich, Dokumentationen, beispielsweise mittels Docstrings und Kommentaren, zu erstellen oder zu vervollständigen. Dies hilft dabei, den Code verständlicher und wartbarer zu machen.

In diesem Fall, explizit Github Copilot betreffend, ermöglichen *Integrierte Prompts,* die mit */doc, /test, /fix und / simplify* abgerufen werden können, spezifische Aufgaben wie das Generieren von Dokumentationen, das Erstellen und Ausführen von Tests sowie das Beheben von Fehlern und Optimieren von Code (Martin und Schiller-Stoff 2023). Weitere Funktionen sind der *@terminal agent*, der bei der Generierung und Ausführung von terminal commands unterstützen kann. Auch commit messages können generiert, sowie Issues durch die Implementierung der closeIssue() Funktion "als bulk" geschlossen werden (Kerr, Kedasha 2024).

AI-Pair-Programmers können zudem auch beim Refactoring unterstützen, indem sie den bestehenden Code analysieren und Optimierungen vorschlagen, die die Lesbarkeit und Wartbarkeit erhöhen. Auch liefern sie Unterstützung bei der Fehlerbehebung, dem Debugging sowie der Codeoptimierung, indem sie Hinweise zur Optimierung der Effizienz und Performance geben.

Mittels der integrierten Chatfunktion können Entwickler:innen direkt in ihrer Entwicklungsumgebung Fragen stellen und spezifische Befehle ausführen, sich den Code

(davon auch einzelne Stellen) erklären oder zusammenfassen lassen. Somit sind Copilots also nicht nur ein Werkzeug zur Steigerung der Produktivität, sondern auch ein Mittel für eine tiefere, kontextuelle Interpretation des Codes, indem sie zu dessen Verständnis beitragen.

Transformation der Digital Humanities: Auswirkungen auf Forschung und zukünftige Potenziale

Es stellt sich die Frage welche Bereiche der DH von den Veränderungen durch den Einsatz von Coding-Assistenten[1] besonders betroffen sind. Beispielsweise könnten jene, die in der *Taxonomy of Digital Research Activities*[2] unter *Creating: Softwareentwicklung* und *Web Development* fallen, genannt werden.

Laut einer Studie von GitHub im Jahr 2022 wurde nachgewiesen, dass der Einsatz von KI-Assistenten in der Softwareentwicklung zu einer Produktivitätssteigerung führte: Programmierer:innen, die unter Einsatz von GitHub Copilot (damals noch nicht allgemein verfügbar) an der Implementierung eines HTTP-Servers arbeiteten, waren im Schnitt etwa 55,8% schneller als ihre Kolleg:innen ohne derartige Hilfe (Peng et al. 2023). Auch in den DH ist von einer gewissen Effizienzsteigerung auszugehen: Die Automatisierung von Routineaufgaben und die Unterstützung bei komplexen Programmierproblemen durch bereits genannte Funktionen ermöglichen es Forschenden in den DH, sich stärker auf die inhaltliche Analyse und Interpretation der Projektdaten zu konzentrieren. Der Vorteil der KI-Assistenten für die Programmierung ergibt sich somit vor allem daraus, dass diese redundante Aufgaben erledigen können (Zhang et al. 2023). Die Annahme eines Focus-Shifts gibt auch eine aktuelle Studie von Hoffmann et al. (Harvard Business School, November 2024) wieder. Aus dieser geht hervor, dass erfahrene Programmierer:innen unter Einsatz von GitHub Copilot ihren Fokus wieder vermehrt auf sogenanntes " *core work"* setzen können, sich aber auch mit mehr *"exploratory work"* und der damit einhergehenden Innovation beschäftigen können. Auch Projektmitarbeiter:innen bzw. Manager:innen, die kaum bis nicht (mehr) programmieren, profitieren von diesen Assistenten, da sie sich schneller ins *"core work"* einarbeiten können (Hoffmann et al. 2024).

Derzeit unterstützt der Vorreiter unter den KI-Assistenten GitHub Copilot mehrere Programmiersprachen und Frameworks. Die beste Leistung zeigt sich jedoch bei *C, C++, C#, Go, Java, JavaScript, PHP, Python, Ruby, Rust, Scala,* und *TypeScript* (Anand et al. 2024). Ähnlich sieht es bei anderen Assistenten aus. Diese Sprachen sind durchaus auch für die DH relevant, jedoch gibt es hier noch Potenziale zu sehen: Eine bessere Unterstützung für SQL (Structured Query Language) sowie Abfragesprachen wie SPARQL (SPARQL Protocol and RDF Query Language) und Standards zur Codierung bzw. Beschreibung von Daten wie RDF (Resource Description Framework) könnte Arbeitsprozesse beschleunigen und die Ergebnisse durch Fehlervermeidung optimieren.

Ein weiterer Bereich, der sich durch die Etablierung von generativer KI wie *ChatGPT*[3] grundlegend verändert hat, ist die Lehre. Nun wird auch die Arbeit mit KI-Assistenten schrittweise in den (Programmier-)Unterricht integriert, denn der Einsatz dieser Tools - und somit eine KI-gestützte Programmierung - wird wohl auch in Zukunft von größerer Relevanz sein. Diverse Workshops[4] bieten auch einer größeren Interessensgemeinschaft Einblicke in diese fortschreitende Entwicklung.

Die Integration von KI-Assistenten in die Software der Digital Humanities

Die Integration von KI-Assistenten in den Tools der DH befindet sich (Stand Juli 2024) noch im Anfangsstadium. Während größere, kommerzielle Firmen und Unternehmen integrierte KI-Assistenten bereits in ihre Produkte integrieren, gibt es im Bereich der Research Software der DH noch viel Luft nach oben:

Aktuell werden diese beinahe ausschließlich als Integration in Entwicklungsumgebungen zur Unterstützung beim Programmieren verwendet. Die Integration kostenpflichtiger Tools wie GitHub Copilot wirft sowohl grundlegende Fragen als auch Kostenfragen auf, die Forschende vor Herausforderungen stellen. Zudem muss hinterfragt werden, inwiefern man sich in den DH auf kommerzielle Software stützen möchte. Grundsätzlich gilt: Je freier zugänglich, desto besser. Alternativ stehen mittlerweile auch Open-Source-Assistenten zur Verfügung. Diese sind jedoch häufig weniger intuitiv und verlangen eine intensive Auseinandersetzung mit der Materie.

Als eine relevante Entwicklung im Bereich der in den DH verwendeten Software kann auch der *Oxygen AI Positron Assistant* genannt werden. Dieses Tool, das bereits im Oktober 2023 veröffentlicht wurde, integriert KI-generierte Inhalte in den Bearbeitungsprozess, indem es Funktionen wie die Verbesserung bestehender Inhalte, Übersetzungen, Fehlerkorrekturen und Vorschläge zur Verbesserung der Lesbarkeit bietet. Problematisch ist wiederum die Tatsache, dass es sich um ein kommerzielles Tool handelt, welches neben der Oxygen-Lizenz für die Benutzung auch einen monatlichen Beitrag verlangt.

Inwiefern und wann sich die Situation des RSE in den DH ändern wird, scheint derzeit aufgrund der Unsicherheit über eine längerfristige Nachhaltigkeit sowie die Leistbarkeit dieser Tools nicht sicher absehbar.

Problematik

Trotz genannter Schwierigkeiten scheinen AI-Pair-Programmers in den DH prävalenter zu werden. Nichtsdestotrotz muss man sich der Limitationen und der mit der Nutzung dieser Tools einhergehenden Problematiken bewusst sein:

KI-Assistenten werden als eine Art Black Box bezeichnet, da die zugrundeliegenden Daten meist unbekannt sind - Das gilt auch für Open-Source-Modelle. Daraus gehen datenschutzrechtliche Bedenken hervor, wie die Intransparenz der Datensammlung und -verarbeitung.[5] Außerdem kann dies auch zur unbewussten Reproduktion bestimmter Verzerrungen im generierten Code führen, welche unbeabsichtigte Diskriminierungen verstärken könnten (Rothlauf et al. 2021).

Zwar scheint die allgemeine Performance dieser Tools über der durchschnittlichen Leistung bzw. Programmiergeschwindigkeit eines Menschen zu stehen, jedoch legen diese kaum Wert auf die Vermeidung von *code smells*, d.h. Schwachstellen im Code, die mit der Zeit zu größeren Problemen führen können, sowie die Einhaltung von Konventionen und *best practices* in der Programmierung. Pudari und Ernst kommen in ihrer Studie (2023, preprint) zu dem Schluss, dass GitHub Copilot bei der Programmierung von Python-Code in etwa 92% der Fälle nicht jene Lösungen als erstes vorschlägt, die den Programmier-Konventionen, den sogenannten *programming idioms*, entspricht - Das obwohl diese am häufigsten von Programmierer:innen genutzt werden und deshalb auch in den Trainings-Datenbanken stark vertreten sein müssten (Pudari und Ernst 2023). Die Abweichung von diesen Standards könnte zu ineffizienten, falschen und sogar unsicheren Code führen. Dies erzeuge Schwachstellen und mache Software anfällig für Angriffe: Bei sogenannten " *Indirect Prompt-Injections*" wird durch das gezielte Einfügen von " *malicious inputs*" in Eingabeaufforderungen KI-Modelle so manipuliert, dass diese unerwünschte und unsichere Ergebnisse liefern. Ein ähnliches Problem ist das " *Data- and Model-Poisoning*", bei welchem die Trainingsdaten (u.a. auf GitHub oder HuggingFace) der KI-Modelle mit falschem und unsicherem Code "vergiftet" werden (Bundesamt für Sicherheit in der Informationstechnik 2024).

Somit erfordert die Korrektur des generierten Codes zusätzlichen Aufwand und nivelliert die Zeitersparnis, die man durch KI-Assistenten scheinbar gewonnen hat. Der Annahme einer wie von Peng et al. statuierten Produktivitätssteigerung stellt sich eine weitere Studie entgegen: *GitClear* (Entwickler:innen des gleichnamigen Tools für die statistische Software-Analyse) analysierte insgesamt 153 Millionen Zeilen GitHub-Code aus dem Zeitraum Januar 2020 bis Dezember 2023 und kam zu der Erkenntnis, dass dessen Qualität seit 2023 kontinuierlich abnahm. Der Grund sei, dass mehr schnell erzeugter und sogenannter "*copy-pasted*"-Code auf der Plattform abgelegt wurde. Dabei handelt es sich laut GitClear eben um KI-generierten oder -veränderten Code. Da Pair-Programmers grundsätzlich nicht vorschlagen, Code zu löschen oder an eine andere Stelle des Programms zu bewegen, käme dieser teilweise doppelt in denselben Projekten vor - was Fragen und Probleme in Bezug auf Nachhaltigkeit und Pflegbarkeit von Software aufwirft (GitClear 2024). In letzter Konsequenz könnte dieser Qualitätsverlust auch DH-Projekte, die sich stark auf Programmierarbeit stützen, betreffen.

Technische Probleme sind aber nur ein Aspekt der vielfältigen Herausforderungen, die eine Auseinandersetzung mit den neuesten technologischen Entwicklungen unumgänglich machen. Blindes Vertrauen in oder die komplette Ablehnung von Digitalisierungstools stellen - allgemein sowie in den Geisteswissenschaften - erhebliche Hürden dar. Dies gilt auch für den Einsatz von KI-Programmierhilfen, da beide Extreme sinnvolles Arbeiten mit diesen Technologien behindern können (Coester und Pohlmann 2021).

Die Etablierung von KI-Assistenten scheint außerdem Einfluss auf die zwischenmenschliche Kommunikationsbereitschaft zu haben: Laut der bereits genannten Studie von Hoffmann et al. (2024) das Verlangen von Kollaboration zwischen Menschen durch die Nutzung von AI-Pair-Programmers zurückzugehen - im Artikel wird dies jedoch als etwas eher Positives gesehen, da damit etwaige "Reibungen" zwischen Projektmitarbeiter:innen umgangen werden (Hoffmann et al. 2024).

Verlässt man sich also zu sehr auf KI-Assistenten, könnte dies möglicherweise die Aufgabe von gesetzten Standards zur Folge haben. Dazu stellen Chun und Elkins (2023) berechtigterweise die Frage: *"Will we program humanity into our tools, or will we cede our humanity to AI?"* Die verstärkte Integration von Künstlicher Intelligenz wirft außerdem Fragen zur zukünftigen Rolle menschlicher Programmierer:innen auf. Es besteht die Möglichkeit, dass KI die Programmierarbeit in den Digital Humanities vollständig übernehmen könnte. In einem Szenario, in dem KI den Großteil der Arbeit übernimmt, wären traditionelle Programmierkenntnisse weniger gefragt, was bedeutende, derzeit unabsehbare Auswirkungen auf den Arbeitsmarkt und die Ausbildung in den DH haben könnte.

Ein ganz anderer Aspekt reiht sich noch bei den Problematiken ein, die die zunehmende Verbreitung und Nutzung von Künstlicher Intelligenz bzw. Coding-Assistenten mit sich bringt: Durch den hohen Energieverbrauch und die da-mit verbundenen CO_2-Emissionen trägt die Nutzung dieser Tools zur Umweltbelastung bei. Besonders rechenintensive KI-Anwendungen, wie das Training großer Sprachmodelle, erfordern beträchtliche Rechenressourcen, die zumeist aus fossilen Energiequellen gespeist werden (Wang et al. 2024). Der Energieverbrauch durch KI wird sich bis 2030 sogar mehr als verdreifachen (Granskog et al. 2024).

Fazit und Ausblick

Der Einfluss informationstechnischer Entwicklungen (Vogeler und Hofeneder 2023) wie der KI-Assistenten auf die Digital Humanities ist nicht zu unterschätzen. Sie bieten erhebliches Potenzial zur Effizienzsteigerung und zur

Erleichterung von Routineaufgaben, wodurch sich Forschende stärker auf die inhaltliche Analyse und Interpretation konzentrieren können. Die Einführung dieser Assistenten hat bereits signifikante Veränderungen vor allem in der Softwareentwicklung der DH bewirkt und könnte diese Transformation weiter vorantreiben. Dennoch sind die Herausforderungen und potenziellen Risiken nicht zu ignorieren: Die Abhängigkeit von kommerziellen Anbietern, die Komplexität der Integration in bestehende Systeme sowie die Gefahr der unbewussten Reproduktion von Verzerrungen oder gar gefährlichem Code durch Black-Box-Modelle sind wesentliche Aspekte, die sorgfältig betrachtet werden müssen. Ein weiteres kritisches Thema ist die zukünftige Rolle menschlicher Programmierer:innen in den DH. Während KI-Assistenten viele Aufgaben übernehmen können, bleibt unklar, wie sich dies langfristig auf die Arbeitsweise, die Ausbildung und den Arbeitsmarkt auswirken wird. Wie bereits erwähnt besteht die Gefahr, dass eine übermäßige Abhängigkeit von KI zur Aufgabe etablierter Standards führen könnte - Hier ist nochmals zu erwähnen, dass die von diversen Seiten angepriesene Produktivitätssteigerung wohl doch auch als zweifelhaft angesehen werden kann, und menschliche Expertise, u.a. in Bezug auf data literacy und best practices, immer noch unverzichtbar ist und sein wird: Die Programmierung ist viel mehr als *programming contests* (Pudari und Ernst 2023), und Digital Humanists nicht als " *code-monkeys*" zu verstehen, deren Ziel es ist, immer schneller immer mehr Code zu produzieren. Die Digital Humanities stehen vor der Aufgabe, mit diesen technischen Entwicklungen Schritt zu halten, dabei aber gängige Standards zu wahren und die Nachhaltigkeit der DH-Software zu garantieren.

Eine kritische Auseinandersetzung mit derartigen Tools wird demnach unumgänglich sein. Das Bundesamt für Sicherheit in der Informationstechnik in Deutschland hat diesbezüglich Empfehlungen[6] für den Einsatz von KI-Programmierassistenten herausgegeben, welche auch die Schwächen der Assistenten berücksichtigen. Dieses Bestreben ist ein Schritt in die richtige Richtung, denn AI-Pair-Programmers sind ein "Double-Edged-Sword" (Zhang et al. 2023): Entwickler:innen in den Digital Humanities werden sich schließlich genau überlegen müssen, wo und wann sie KI-Assistenten in Zukunft einsetzen werden.

Fußnoten

1. Der Begriff "Copilot" wird umgangssprachlich für AI-Pair-Programmers verwendet, ist jedoch stark durch GitHub Copilot geprägt. Angesichts der zahlreichen Alternativen erscheint eine neutrale Bezeichnung angemessener.
2. TaDiRAH Browser: https://www.tadirah.info/pages/Browser.html
3. https://openai.com/chatgpt/
4. Siehe dazu die Workshopreihen "AI assistance systems in the digital humanities and research software engineering", https://zenodo.org/communities/aiassistingdh/records?q=&l=list&p=1&s=10&sort=newest; "Angewandte Generative KI in den Digitalen Geisteswissenschaften" der AGKI-DH, https://agki-dh.github.io/index.html
5. Siehe "FAQ zum Thema KI und Datenschutz": https://www.dsb.gv.at/download-links/FAQ-zum-Thema-KI-und-Datenschutz.html.
6. Siehe dazu "Ai Coding Assistans", Bundesamt für Sicherheit in der Informationstechnik, 2024, Link: https://www.bsi.bund.de/DE/Service-Navi/Presse/Alle-Meldungen-News/Meldungen/ANSSI_BSI_KI-Programmierassistenten_241004.html?nn=520690.

Bibliographie

Anand, Megha, Gordon Hogenson, Amy Nguyen, Beth Harvey, Saisang Cai, Jill Grant, Huaping Yu, Terry G. Lee, J. Martens, Tom FitzMacken. 2024. "Informationen zu GitHub Copilot-Abschlüsse in Visual Studio - Visual Studio (Windows)", 22. Mai 2024. https://learn.microsoft.com/de-de/visualstudio/ide/visual-studio-github-copilot-extension?view=vs-2022 (zuletzt zugegriffen am 10. Juli 2024).

Bundesamt für Sicherheit in der Informationstechnik. 2024. "AI Coding Assistants". Deutsch-Französische Empfehlungen für den Einsatz von KI-Programmierassistenten. Last updated September 2024. https://www.bsi.bund.de/DE/Service-Navi/Presse/Alle-Meldungen-News/Meldungen/ANSSI_BSI_KI-Programmierassistenten_241004.html?nn=520690. (zuletzt zugegriffen am 7. November 2024).

Chun, Jon und Katherine Elkins. 2023. The Crisis of Artificial Intelligence: A New Digital Humanities Curriculum for Human-Centred AI, International Journal of Humanities and Arts Computing 17.2 (2023): 147–167, https://doi.org/10.3366/ijhac.2023.0310.

Coester, Ulla, und Norbert Pohlmann. 2021. „Vertrauen – ein elementarer Aspekt der digitalen Zukunft". Datenschutz und Datensicherheit - DuD 45 (2): 120–25. https://doi.org/10.1007/s11623-021-1401-x.

GitClear. 2024 „Coding on Copilot: 2023 Data Suggests Downward Pressure on Code Quality (incl 2024 projections). https://www.gitclear.com/coding_on_copilot_data_shows_ais_downward_pressure_on_code_quality.(zuletzt zugegriffen am 12. November 2024).

Granskog, Anna, Sofia von Schantz, Diego Hernandez Diaz, Jesse Noffsinger, Lorenzo Moavero Milanesi, Pankaj Sahcdeva, Arjita Bhan, Michiel Nivard. 2024. „Unlocking the European AI revolution". McKinsey. 24. Oktober 2024. https://www.mckinsey.com/industries/electric-power-and-natural-gas/our-insights/the-role-of-power-in-unlocking-the-european-ai-revolution. (zuletzt zugegriffen am 7. November 2024).

Hoffmann, Manuel, Sam Boysel, Frank Nagle, Sida Peng, und Kevin Xu. 2024. „Generative AI and the Nature of Work", November 2024. https://doi.org/10.2139/ssrn.5007084.

Kerr, Kedasha. 2024. „10 Unexpected Ways to Use GitHub Copilot“. The GitHub Blog (blog). 22. Januar 2024. https://github.blog/2024-01-22-10-unexpected-ways-to-use-github-copilot/ (zuletzt zugegriffen am 10. Juli 2024).

Martin, Julia, und Sebastian Schiller-Stoff. 2023. „Copilot technologies in the digital humanities“. Dezember 7. https://doi.org/10.5281/zenodo.10288101.

Pudari, Rohith, und Neil A. Ernst. 2023. „From Copilot to Pilot: Towards AI Supported Software Development“. arXiv. https://doi.org/10.48550/arXiv.2303.04142.

Peng, Sida, Eirini Kalliamvakou, Peter Cihon, und Mert Demirer. 2023. „The Impact of AI on Developer Productivity: Evidence from GitHub Copilot“. arXiv. 13. Februar 2024. http://arxiv.org/abs/2302.06590.

Rothlauf, Franz, Dominik Sobania und Martin Briesch. 2021. „Choose Your Programming Copilot: A Comparison of the Program Synthesis Performance of GitHub Copilot and Genetic Programming“. arXiv (Cornell University), November. https://www.academia.edu/118312983/Choose_Your_Programming_Copilot_A_Comparison_of_the_Program_Synthesis_Performance_of_GitHub_Copilot_and_Genetic_Programming.

Schiller-Stoff, Sebastian, und Julia Martin. 2024. „Legacy Code and Copilot Technologies: AI-Supported Software Maintenance“. Januar 22. https://doi.org/10.5281/zenodo.10550479.

Schiller-Stoff, Sebastian, und Leona Elisabeth Münzer. 2024. „IDE-Integration of GitHub Copilot: Jetbrains IDEs“. Mai 15. https://doi.org/10.5281/zenodo.11196383.

Vogeler, Georg, und Philipp Hofeneder. 2023. „Die digitale Transformation der österreichischen Geisteswissenschaften und ihre Herausforderungen für die Zukunft: Strukturelle Perspektiven für die Integration von Datenwissenschaften, maschinellem Lernen und künstlicher Intelligenz“. Zeitschrift für Hochschulentwicklung 18 (Sonderheft Forschung): 141–162. https://doi.org/10.21240/zfhe/SH-F/09.

Wang, Qiang, Yuanfan Li, und Rongrong Li. 2024. „Ecological footprints, carbon emissions, and energy transitions: the impact of artificial intelligence (AI)“. Humanities and Social Sciences Communications 11, 14. August 2024. https://doi.org/10.1057/s41599-024-03520-5.

Zhang, Beiqi, Peng Liang, Xiyu Zhou, Aakash Ahmad, und Muhammad Waseem. 2023. Practices and Challenges of Using GitHub Copilot: An Empirical Study. https://doi.org/10.18293/SEKE2023-077.