Incorporating Generative AI into Software Development Education

Olga Petrovska
olga.petrovska@swansea.ac.uk
Swansea University
Swansea, Wales, UK

Lee Clift
l.a.clift@swansea.ac.uk
Swansea University
Swansea, Wales, UK

Faron Moller
f.g.moller@swansea.ac.uk
Swansea University
Swansea, Wales, UK

Rebecca Pearsall
rebecca.pearsall@ou.ac.uk
The Open University
Cardiff, Wales, UK

ABSTRACT

This paper explores how Generative AI can be incorporated into software development education. We present examples of formative and summative assessments, which explore various aspects of ChatGPT, including its coding capabilities, its ability to construct arguments as well as ethical issues of using ChatGPT and similar tools in education and the workplace. Our work is inspired by the insights from surveys that show that the learners on our Degree Apprenticeship Programme have a great interest in learning about and exploiting emerging AI technology. Similarly, our industrial partners have a clear interest for their employees to be formally prepared to use GenAI in their software engineering roles. In this vein, it is proposed that embedding the use of GenAI tools in a careful and creative way - by developing assessments which encourage learners to critically evaluate AI output - can be beneficial in helping learners understand the subject material being taught without the risk of the AI tools "doing the homework".

CCS CONCEPTS

• **Applied computing** → **Education**; • **Social and professional topics** → **Software engineering education**; • **Computing methodologies** → *Artificial intelligence.*

KEYWORDS

generative AI, software engineering, education, apprenticeship, assessment

ACM Reference Format:
Olga Petrovska, Lee Clift, Faron Moller, and Rebecca Pearsall. 2024. Incorporating Generative AI into Software Development Education. In *Computing Education Practice (CEP '24), January 5, 2024, Durham, United Kingdom.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3633053.3633057

CEP '24, January 5, 2024, Durham, United Kingdom

ACM ISBN 979-8-4007-0932-6/24/01.
https://doi.org/10.1145/3633053.3633057

1 INTRODUCTION

After many decades of promise, AI has recently been – visibly and publicly – rapidly integrating itself more and more into various aspects of modern life, transforming the way we work and learn. From simple predictive text based on Markov chains, we have moved on to far richer Generative AI (GenAI) tools, complex neural network models like Generative Pre-trained Transformers (GPTs) which produce increasingly complex outputs. In particular, since its release at the end of 2022, OpenAI's ChatGPT has undergone rapid growth and widespread adoption. According to Deloitte's Digital Consumer Trends 2023 survey [12], 52% of people in the UK are aware of such GenAI tools, and half of these have used it. Disturbingly, 43% of those who used it assume outright that the information it provides is factual and 38% consider it to be unbiased.

Software engineering is no exception to the current adoption trends. AI has been reshaping the software industry for some time now. It enables software engineers to simplify their work by automating repetitive tasks, improving the debugging process, streamlining testing, and providing a multitude of other functionalities. Increased accessibility of GenAI tools means there is even more to gain from it in the day-to-day life of a software engineer. However, this raises concerns about over-reliance on this technology without sufficient understanding of its capabilities and limitations.

With this in mind, we explore here the feasibility of incorporating GenAI tools into software engineering education. We also present the ideas for assessments aimed at raising awareness about the capabilities and limitations of GenAI. These are experimental assessments that we carried out in our Applied Software Engineering Degree Apprenticeship Programme.

2 BACKGROUND

Recently, a number of research papers have appeared investigating the abilities of LLMs (large language models) to generate executable code for various languages [3, 5, 9, 10, 13, 19]. For example, Hendryk et al. [9] introduced APPS, a benchmark for code generation. Khan and Uddin [10] investigated GPT's performance in generating code documentation for Java, Python, PHP, GO, JavaScript, and Ruby. Peng et al. [13] explored the impact of GitHub Copiloton software developer's productivity, observing a 55.8% reduction in development time in their controlled experiment.

Various educators and researchers have looked into how GenAI – and ChatGPT in particular – impacts education both in terms of the

learning process and assessment [11, 14, 16–18, 20]. Finnie-Ansley et al. [6] found that OpenAI's Codex solves the majority of CS2 programming exercises in Python. Denny et al. [4] investigated Copilot's performance solving CS1 Python programming tasks and found that it solved half of them on its very first attempt. Biderman and Raff [2] explored the performance of Java code generated by GPT-J on MOSS (measure of software similarity tool) and discovered that plagiarism was not detected, although they do not have evidence that this would be the case for tasks in more advanced programming courses.

Barke et al. [1] conducted a study with 20 programmers to explore how they interact with Copilot, and observed occasional over-reliance of first-time users. Prather et al. [15] explored interactions of novice programmers with Copilot. Zastudil et al. [21] interviewed computer science students and instructors to understand their interaction with GenAI and their expectations regarding its use in education. They observed a positive attitude with regard to integrating GenAI into the educational process as well as concerns about over-reliance, trustworthiness and academic integrity.

In terms of education policy, there have been numerous discussions at a national level exploring the use of ChatGPT in Higher Education in the UK. For example, the QAA (Quality Assurance Agency) organised a series webinars addressing assessment in the era of ChatGPT [8]; and the Department of Education submitted a call for evidence regarding GenAI in education [7].

3 METHODOLOGY

For our study, we followed a structured approach to collecting data by designing three anonymous online surveys aimed at three distinct groups: first-year learners (12 respondents); final-year learners (12 respondents); and industry professionals (13 respondents). Following the surveys, we ran an experimental summative assessment for the first-year learners as a part of a Professional Issues module.

Professionals Survey. This survey was designed to identify how much awareness businesses and organisations have about GenAI, and to gain insights into industry's perspective surrounding the integration of GenAI technology into their work processes. We also aimed to understand whether businesses are interested in their employees receiving training to utilise these new technologies.

Learner Surveys. Learner surveys were designed to run alongside experimental formative assessments. Both first- and final-year learners were asked to answer background-related questions to assess their familiarity with GenAI and the extent to which they carry out programming tasks on a daily basis. Similarly, both cohorts were asked to express their thoughts on how and whether GenAI should be used in software development, and to estimate potential time savings from the pre-generated output.

Formative Assessments. As a part of their Professional Issues module, first-year learners analysed a pre-generated ChatGPT output in Java for a Stack class, comparing it to their own code previously written as part of their formative assessment in another module. They were asked to answer a number of open-ended and multiple choice questions as well as evaluate the quality of the pre-generated code using a 5 point Likert scale.

As a part of their formative assessment for an AI module, final-year learners were asked to use their own prompts to ChatGPT to generate a Dart application for a Snooker Scoring app. They were then asked to analyse the output with respect to their own solution which they had done for another module, and rate it on a Likert scale. They had also to mark the output based on the original marking rubric for the coursework, using standard grade boundaries; and estimate time savings associated with the use of ChatGPT.

Summative Assessment. An experimental summative assessment was created to evaluate first-year learners' understanding of ChatGPT. The cohort completing this assessment consisted of 17 learners. The assessment looked not only into the analysis of ChatGPT's programming capabilities but also into argumentation and ethical issues. The results of the assessment were analysed through the creation and use of a coding scheme, to gauge sentiment towards different tasks, and GenAI usability.

The first two parts of the assessment required interacting with ChatGPT to generate well-formatted and functioning code in Java and Pep/8 Assembly, respectively. The Java task was similar to that in the previous formative assessment; however, this time, learners were picking their own prompts and had an opportunity to refine their prompts to get a better result. They were also asked to reflect on their 'journey'. Pep/8 Assembly was chosen for the second task as it is covered in a previous module, and because it adds confusion to prompts: as PEP8 is a style guide for Python code, if assembly language is not explicitly specified, ChatGPT will likely generate Python code. Additionally, numerous tests that we ran on ChatGPT showed that, compared to simple Java snippets, the code that it generates in Pep/8 has significantly more issues.

In the third part of the assessment, the learners were asked to write a prompt for ChatGPT to argue that cryptocurrencies are (environmentally) sustainable. They were also tasked to explore a topic of their choice, asking ChatGPT to argue that "A is better than B" and then argue the inverse. Reflections that learners documented allowed an evaluation of how much awareness they have regarding ChatGPT's potential biases and delusions. The marking rubric for each of the above parts considers how well prompts are constructed, the rigour with which the output is analysed and compared with the learner's own work, and the quality of their 'journey' presentation.

Finally, in the last part, learners were asked to discuss the implications of using ChatGPT in education and in their workplace, considering both professional and ethical aspects. The marking rubric for this part takes into account the thoroughness of the learner's evaluation of the overall experience, both with programming and with text output. It also considers the comprehensiveness of their ethical analysis.

4 RESULTS

First-year Survey and Formative Assessment. All but two respondents agreed that ChatGPT generated a working Java program requiring little to no modification. Interestingly, there was no correlation between people's exposure to software development and the way in which the output was rated. ChatGPT's output was described as *"more efficient and better styled"*, though respondents also noted that it: lacked *"some common sense like variable naming"*;

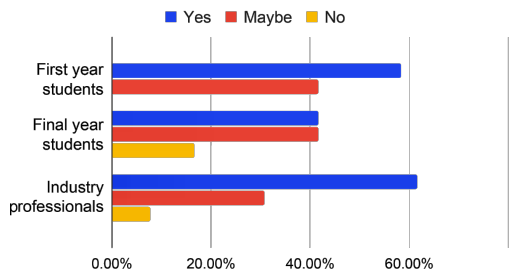

Figure 1: Responses to "Do you feel you / your staff being formally trained on GenAI (e.g., ChatGPT) as part of this Degree Apprenticeship Programme would be beneficial?"

did not include comments or follow coding conventions; and did not provide input validation.

When asked about the benefits of ChatGPT as a support tool, half agreed that it would have been beneficial as a support tool in their programming module, while a third opted for the 'maybe' option. Improved efficiency, new ideas and solutions, and a pre-generated 'skeleton' code were mentioned among the benefits.

Final-year Survey and Formative Assessment. Most of the respondents managed to get a running program of varied quality, with 50% of the respondents stating that it needed heavy modifications. Everyone felt that the generated output was worse than their own previously developed app and that ChatGPT struggled with specific features, requiring further prompts to improve the code. At the same time, everyone believed that they would have benefitted from using ChatGPT during their coursework assignment, specifically by saving time. They believe it would be useful for refactoring or writing initial small bits of code, but not a whole program.

In terms of time-savings, we asked participants to estimate the amount of time that they spent on the programming task initially, and to provide an estimate on how long it would take them if they were to use ChatGPT. The average development time without ChatGPT was estimated to be 17.1 hours, and the estimate for development time with ChatGPT was 10 hours.

Three-quarters of the final-year learners feel that ChatGPT would be useful in their full-time job by saving time and offering alternative solutions to programming tasks.

Industry Professionals Surveys. With respect to GenAI in an industry setting, only two of the 13 respondents stated that there were attempts to integrate GenAI into their workplace. This can be explained by the fact that this technology has only entered a mass market recently. However, there is a clear trend of seeing GenAI as useful in improving a work life cycle, with 30.8% of the respondents considering it definitely useful and 61.5% of the respondents considering it potentially useful.

Combined Overall Survey Results: GenAI Education and Practice. A majority of the surveyed participants see a benefit in formal training on the use of GenAI tools as an aid in software engineering. Figure 1 shows that over 80% of all participants demonstrated some level of willingness to either want to be trained or to have their staff formally trained.

Summative Assessment. As the first part of the summative assessment, first-year learners were asked to solve a simple Java task, something they had completed earlier in the year for another module's formative assessment. Unlike formative assessment, which used a pre-generated code, this assignment required them to interact with ChatGPT. When generating Java code via ChatGPT, 94% of learners generated code with their first prompt which compiled, and 89% of them felt that it was equal to or better than the code they wrote themselves in their attempt.

When refactoring their generated code, 53% of learners felt that it refactored correctly, and ChatGPT could implement additional features, or modify existing ones. In other cases, code either did not compile after refactoring or learners felt that the newly refactored code didn't reflect the prompt they had given. Due to ChatGPT being a generalised LLM, this behaviour is not surprising and can become more common the more complex the refactoring request.

Compared to generating Java code, the learners struggled to get ChatGPT to generate Pep/8 Assembly code. They were tasked with generating code which, when provided two integers, will divide one by the other; something which is tricky in Pep/8. Results show that 29% of learners were able to generate some compilable Pep/8 code, although no learner felt that the generated code was better than what they could write themselves, and only 2 learners (11%) felt that it was equal to what they would write. The remaining 71% of learners either couldn't generate compilable Pep/8 code or their prompts were not specific enough and generated Python code instead. In fact, one of the learners even failed to notice that the code was not in Pep/8, which is an example of over-reliance.

When tasked with generating a text-based argument regarding the sustainability of cryptocurrencies, most learners (76%) were critical of the arguments made by ChatGPT. Many of them reported that the arguments made were either vague or that it wasn't a balanced discussion, and the generated discussion was one-sided. After requesting refinement of the argument, more learners (81%) were critical of the newly generated response, either due to exasperating previous issues or creating new ones. Many learners felt that the generated text wasn't sufficient for a constructive argument, and would require additional work and editing. From this response, it can be understood that many learners may not feel comfortable to use ChatGPT for large amounts of nuanced text generation, as it is reported to do a worse job than what many of them could do by themselves.

When asked to reflect on their experience, 62.5% of learners felt that ChatGPT could make either limited or well-reasoned factual arguments, in many cases providing reasonable references and discussing a topic in depth. However, many learners felt that the writing style was very artificial. Additionally, 68% of learners felt that ChatGPT was able to generate reasonable code, in the aforementioned languages, as well as C# or Javascript (Figure 2). With such a large majority of learners finding code generation useful, this supports the argument for using GenAI as a teaching aid.

Further support for this can be seen when analysing learners' thoughts towards using GenAI in work. When asked if they felt that GenAI should be used within the workplace, 62.5% responded positively, stating they feel that when used correctly, this could increase productivity and make code generation easier, a similar statistic to the one seen in the previously conducted surveys. All

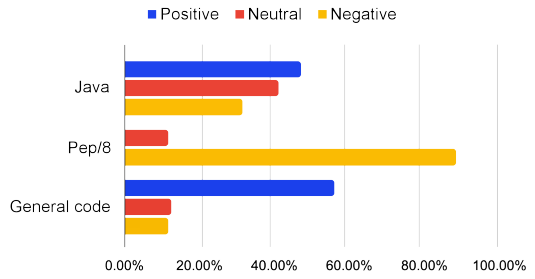

Figure 2: Sentiment towards language code generation

37.5% of learners who felt that it wouldn't be useful in the workplace had previous neutral or bad experiences with code generation, either feeling that the code generated in the initial Java task was equal to their own or feeling that ChatGPT failed to refactor their Java code effectively. Almost all learners had a level of concern when considering using GenAI in the workplace, mostly around privacy and ethical considerations.

5 CONCLUSION AND FUTURE WORK

The purpose of this paper was to describe our experience of incorporating GenAI into teaching, which helped share with students a new tool they can access when creating software. Additionally, we hope to motivate further discussion to set guidelines for how to incorporate this new technology into the curriculum. Results show that when employed in formative and summative assessments, students can use GenAI to complete tasks, and can easily identify positives and negatives regarding this new technology.

Apprentices provided positive feedback, overall, at being allowed to explore technologies such as GenAI within the curriculum. They expressed enjoyment in discussing and using GenAI in their summative assessments, and – especially the more reluctant users – applauded the opportunity to explore technology which they had either dismissed or felt was beyond their skill level.

Future work involves incorporating GenAI into more modules on the Degree Apprenticeship Programme, further preparing learners for an ever-changing technology landscape. Specifically, we are working on creating summative assessments which give students the choice of using these tools, further normalising them as another 'tool in the toolbox' for software engineers.

REFERENCES

[1] Shraddha Barke, Michael B. James, and Nadia Polikarpova. 2023. Grounded Copilot: How Programmers Interact with Code-Generating Models. *Proceedings of the ACM on Programming Languages* 7 (April 2023). Issue OOPSLA1. https://doi.org/10.1145/3586030

[2] Stella Biderman and Edward Raff. 2022. Fooling MOSS Detection with Pretrained Language Models. In *CIKM '22: Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, Vol. 11. ACM, Atlanta, GA, 2293–2943. https://doi.org/10.1145/3511808.3557079

[3] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating Large Language Models Trained on Code. (2021). arXiv:2107.03374 [cs.LG]

[4] Paul Denny, Viraj Kumar, and Nasser Giacaman. 2023. Conversing with Copilot: Exploring Prompt Engineering for Solving CS1 Problems Using Natural Language. *SIGCSE 2023 - Proceedings of the 54th ACM Technical Symposium on Computer Science Education* 1 (3 2023), 1136–1142. https://doi.org/10.1145/3545945.3569823

[5] Giuseppe Destefanis, Silvia Bartolucci, and Marco Ortu. 2023. A Preliminary Analysis on the Code Generation Capabilities of GPT-3.5 and Bard AI Models for Java Functions. (2023). arXiv:2305.09402 [cs.SE]

[6] James Finnie-Ansley, Paul Denny, Andrew Luxton-Reilly, Eddie Antonio Santos, James Prather, and Brett A. Becker. 2023. 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* (Melbourne, VIC, Australia) *(ACE '23)*. ACM, New York, NY, USA, 97–104. https://doi.org/10.1145/3576123.3576134

[7] Department for Education. 2023. *Generative artificial intelligence in education Call for evidence.* Retrieved October 5, 2023 from https://consult.education.gov.uk/digital-strategy/generative-artificial-intelligence-in-education/

[8] Quality Assurance Agency for Higher Education. 2023. *ChatGPT and artificial intelligence: QAA webinar series on ChatGPT.* Retrieved August 28, 2023 from https://www.qaa.ac.uk/membership/membership-areas-of-work/academic-integrity/chatgpt-and-artificial-intelligence

[9] Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. 2021. Measuring Coding Challenge Competence With APPS. (2021). arXiv:2105.09938 [cs.SE]

[10] Junaed Younus Khan and Gias Uddin. 2023. Automatic Code Documentation Generation Using GPT-3. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering* (Rochester, MI, USA) *(ASE '22)*. ACM, New York, NY, USA, Article 174, 6 pages. https://doi.org/10.1145/3551349.3559548

[11] Weng Marc Lim, Asanka Gunasekara, Jessica Leigh Pallant, Jason Ian Pallant, and Ekaterina Pechenkina. 2023. Generative AI and the future of education: Ragnarök or reformation? A paradoxical perspective from management educators. *The International Journal of Management Education* 21 (July 2023), 100790. Issue 2. https://doi.org/10.1016/J.IJME.2023.100790

[12] Deloitte LLP. 2023. *Digital Consumer Trends 2023.* Retrieved October 3, 2023 from https://www2.deloitte.com/uk/en/pages/technology-media-and-telecommunications/articles/digital-consumer-trends.html

[13] Sida Peng, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. 2023. The Impact of AI on Developer Productivity: Evidence from GitHub Copilot. (2023). arXiv:2302.06590 [cs.SE]

[14] Renana Peres, Martin Schreier, David Schweidel, and Alina Sorescu. 2023. On ChatGPT and beyond: How generative artificial intelligence may affect research, teaching, and practice. *International Journal of Research in Marketing* 40, 2 (2023), 269–275. https://doi.org/10.1016/j.ijresmar.2023.03.001

[15] James Prather, Brent N. Reeves, Paul Denny, Brett A. Becker, Juho Leinonen, Andrew Luxton-Reilly, Garrett Powell, James Finnie-Ansley, and Eddie Antonio Santos. 2023. "It's Weird That It Knows What I Want": Usability and Interactions with Copilot for Novice Programmers. *ACM Trans. Comput.-Hum. Interact.* (Aug 2023). https://doi.org/10.1145/3617367

[16] Junaid Qadir. 2023. Engineering Education in the Era of ChatGPT: Promise and Pitfalls of Generative AI for Education. In *2023 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, Salmiya, Kuwait. https://doi.org/10.1109/EDUCON54358.2023.10125121

[17] Sangho Suh and Pengcheng An. 2022. Leveraging Generative Conversational AI to Develop a Creative Learning Environment for Computational Thinking. In *27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22 Companion)*. ACM, New York, NY, USA, 73–76. https://doi.org/10.1145/3490100.3516473

[18] Ahmed Tlili, Boulus Shehata, Michael Agyemang Adarkwah, Aras Bozkurt, Daniel T. Hickey, Ronghuai Huang, and Brighter Agyemang. 2023. What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. *Smart Learning Environments* 10 (December 2023), 1–24. Issue 1. https://doi.org/10.1186/s40561-023-00237-x

[19] Immanuel Trummer. 2022. CodexDB: Generating Code for Processing SQL Queries using GPT-3 Codex. *Proceedings of the VLDB Endowment* 15 (2022), 2921–2928. Issue 11. https://github.com/itrummer/CodexDB.

[20] Kailas Vodrahalli, Roxana Daneshjou, Tobias Gerstenberg, and James Zou. [n. d.]. Do Humans Trust Advice More if it Comes from AI? An Analysis of Human-AI Interactions. 22 ([n. d.]), 2022. https://doi.org/10.1145/3514094.3534150

[21] Cynthia Zastudil, Magdalena Rogalska, Christine Kapp, Jennifer Vaughn, and Stephen MacNeil. 2023. Generative AI in Computing Education: Perspectives of Students and Instructors. (2023). arXiv:2308.04309 [cs.HC]