Effect of Large Language Model Use on Programming Project Groups

Laura Graf

Technische Universität München, Arcisstraße 21, 80333 Munich, Germany

Abstract

The adoption of Large Language Models (LLMs) in education has prompted questions about their impact on programming projects. This research will explore how the use of LLMs affects learning and socio-affective outcomes on individual and group level in first software engineering projects. Existing literature explores both potential benefits and pitfalls of LLMs in educational contexts. LLMs enhancing readability, explaining others' code and providing quick answers to less experienced students could improve group work. However, there are concerns such as students' judgment of competency, effort and contributions created with LLM support affecting group collaboration dynamics. To address the gap in empirical research on LLMs' impact on perceptions of teammate competency, connectedness, self-efficacy, learning gain, and professional identification we will analyze not only self-reported measures but also work with process data from collaborative coding platforms to extract meaningful measures of collaboration behavior and issues in group code prominent when LLMs are being used.

Keywords

Large Language Models, Computer Science Education, Collaborative learning, Process Mining, Socio-affective measures

1. Introduction

After the vast adoption of Large Language Models (LLMs) universities have raised questions about the adequacy of curriculum and assessment in response to student use of computer-generated output during their studies (Kasneci et al., 2023). These questions are particularly pertinent for future programming professionals, since LLMs are effective in generating code chunks (Kazemitabaar et al., 2023). The use of LLM-generated code can not only speed up programming tasks but substantially offload thinking processes to the machine. In light of potential automation of coding tasks, it has become unclear as to what skills should be taught to future programming professionals to enable effective integration of LLMs into the human-led process.

The focus on optimization of productivity enabled by machines has thus far been central to research on how to integrate LLMs into human cognitive practices (Wang et al., 2019). Yet such a focus is only partially relevant in educational settings. Educational outcomes target students' cognitive development and higher-order thinking in relation to the domain they study. Tools supporting cognitive processes can benefit the learner in offloading some parts of such a process and allowing the learner to focus on higher- order thinking (Salomon, 2003). At the same time, improper use of the tool can lead to a reduced, shallow understanding (ibid.). Therefore, it is important to understand the relationship between student use of cognitive tools such as LLMs and learning outcomes related to their domain knowledge to ensure that integration does not deprive students from developing much needed higher order thinking.

Another reason why a focus on solely optimizing student use of LLMs is insufficient in educational settings is that technologies continue to evolve. The models change to support humans better, and this process cannot be expected to stabilize on a certain pattern (Joksimovic et al., 2023).

Students will need to continue working together, solving problems and communicating effectively, regardless of the specific cognitive tool they may use. Hence, it is also important to understand the relationship between the use of cognitive tools and educational outcomes that reach beyond domain knowledge. Frequent use of tools like LLMs may shape students in a way that affects them profoundly, so stakeholders need a clearer understanding of how broader educational outcomes are affected by those tools so that instructional practices can be adapted to preserve the focus on developing skills essential for humans.

To address this pressing need, my project will investigate how LLMs affect educational outcomes in a collaborative setting where future programming professionals practice a broad set of skills. Collaborative work is commonly part of software engineering projects in computer science curricula. Collaborative work is an essential part of professional software engineering and interpersonal skills are among the most significant for the effectiveness of software engineers (Boyatzis et al., 2017). Groups have been shown to innovate faster, identify mistakes more quickly, and find better solutions

Proceedings of the Doctoral Consortium of the 19th European Conference on Technology Enhanced Learning, September 16-20, 2024, Krems, Austria

l.graf@tum.de (L. Graf)

to problems; all while reporting a higher job satisfaction (Duhigg, 2019).

Moreover, project-based assignments where students practice collaborative work are a catalyst for bonding and social learning, facilitating social capital among future professionals and affecting professional identity. Collaboration experiences can also create precedents for exclusion and negatively affect belonging and diversity in STEM (Miller-Young et al., 2023). This richness of educational outcomes makes collaborative programming assignments a suitable context to examine the effect of cognitive tools, such as LLMs.

Current literature is limited in explaining the effects that LLMs can have on broader educational outcomes in programming group work. Existing research suggests that such effects could be both positive and negative. LLMs have the potential to support participation of students with less programming skills in code production and help to understand others' code, important for positive collaborative work. However, it could also amplify issues of unequal effort distribution through the option to auto-generate code, which is known to create negative experiences (Nguyen et al., 2023). LLMs can, for instance, facilitate shared understanding by improving code readability and documentation as well as reduce the need for any group members to spend large parts of the time on lower-level tasks such as generating test cases, which might change previously common role distributions. With LLMs affecting most parts of the programming projects, an influence on the social aspects of group work is likely and deserves attention.

This research gap calls for empirical examination of the effects of LLMs in programming assignments on domain- specific knowledge and the effect on group processes, as well as longer-term imprint on students, such as the formation of professional identity. My thesis will focus on addressing this gap. I will employ mixed methods research design. First, I will analyze the effect of LLM integration into programing group projects, in relation to student perceptions of learning, their socio-affective attitudes towards teammates and their evolving identification with the domain. Second, I will investigate the relationship between these perceptions and process data from code progression, as student perceptions are largely mediated by the code-based communication on GitHub. The thesis is in its planning stage.

2. Related Work

When students use LLM-based tools for their tasks in a course, this affects how and what they learn. To advance the goal of understanding the effect of LLMs in collaborative programming assignments, this section explains how LLM use in collaborative programming tasks may affect educational outcomes. This includes individual outcomes, such as learning, self- efficacy, and professional identity, as well as group-related socio-affective outcomes, such as trust in teammates competency and connectedness. I also explain why the process of how learners collaborate when they individually use LLMs must be considered.

2.1. Effect of LLMs on Learning Individuals

LLMs offer a diverse range of applications for enhancement of learning experiences, personalized to the student (Kasneci et al., 2023). In introductory programming assignments, LLM-based coding tools currently already perform at the level that outscores the average student (Finnie- Ansley et al., 2022). Most students prefer using a LLM, especially to get a starting point, even when they often face difficulties in understanding, editing, and debugging generated code (Vaithilingam et al., 2022). Studies have also shown that programmers tend to defer tasks related to comprehension to the LLM, even though this can steer them in the wrong direction (Nam et al., 2024). Some scholars also suggests that students use LLMs for requesting explanations of code and general questions more often than for code generation (Kazemitabaar et al., 2024).

To circumvent challenges associated with LLM use, chatbots have been developed to offer hints to mimic human tutoring, instead of giving students full solutions (Bassner et al., 2024).

Literature so far has shown that integrating LLMs into practices around learning and studying can affect individual learning gains. A major concern here is that when students regularly offload to technology, they may not actually learn how to perform the task on their own. (Darvishi et al. 2024) found that when using LLM, students seem to be finishing tasks well, but once the LLM was removed, they did not replicate the new strategies used by the LLM that were helpful with the tasks. Another study showed that learning gains from using an LLM in learning programming languages vary with context and task complexity (Aviv et al., 2024). Researchers observed that LLMs did not reduce metacognitive difficulties for students with limited programming abilities and even introduced new ones (Prather et al., 2024).

In addition to learning gains, integrating LLMs into learning practices can impact self-perceptions, such as self- efficacy and professional identity. Studies on LLMs' effects on students' self-efficacy found that LLM-supported review of course topics improved students' self- efficacy and motivation (Lee et al., 2022). This effect on self-efficacy appeared because LLM helped students become active during learning, as it provided a safe way to explore questions (Y.-F. Lee et al., 2022). A study where interactions with an LLM supported student thinking deeply about a topic showed improved self-efficacy and learning achievements (Chang et al., 2022). (Wang et al., 2023) found that AI based on good technology combined with technological skills in a higher education program improve students' self-efficacy, mediating performance. Perception of self-efficacy can also benefit from having a starting point in coding (Vaithilingam et al., 2022).

Self-efficacy further plays an important role in securing diversity, equity, and inclusion in STEM. Minorities and women feel less included in the engineering groups in general, but female students' who plan to persist in this male-dominated domain also show high self-efficacy (Marra et al., 2009). It therefore also may be important to ensure that integrating LLMs into collaborative work, where many of the exclusionary practices occur (e.g. William M. Hall, Toni Schmader, Elizabeth Croft, 2015), maintains positive impact on long- term professional orientation, mediated by group experiences.

2.2. Effect of LLMs on Learning Groups

When it comes to collaborative learning settings as in group programming assignments, LLM use reaches beyond the effects on the individual, such as learning, self-perceptions, and future identification. Both socio-cultural (Vygotsky, 1978) and socio-cognitive theories of learning (Dillenbourg, 1990) highlight the influence of the environment on learning, often enacted through peer interactions. LLM use by individual learners can potentially influence peer interactions, mediated by technology, and further impact group- related socio-affective outcomes, such as trust in group members and connectedness.

Research on the use of LLMs in collaborative learning has been limited to the development of tools that target collaborative processes at the group-level. For example, Kasneci et al. (2023) speculate that these tools can facilitate group discussions by providing feedback and personalized guidance to students to improve group participation or give editing recommendations to support collaborative writing. LLMs could help avoid common faults in the group processes by integrating information or promoting knowledge convergence and decision-making – all group-level processes essential for effective collaboration (Westby & Riedl, 2023; Järvelä & Hadwin, 2013; Khakurel & Blomqvist, 2022). It is noteworthy that many of these existing propositions are limited to the LLM-based tools specifically designed to support group work. However, group members can also choose to use LLMs for individual needs, rather than to support group processes. The effects of such individual use within a collaborative task have not yet been explored.

Previous research suggests that AI can affect collaboration in unintended ways (Wang et al., 2022), and this can also be expected when students integrate LLMs to support individual programming needs within a collaborative task. For instance, group-related socio-affective outcomes, such as trust in teammates' competency and feeling of connectedness with the group, may be affected. Students' perceptions of teammates' contributions may change when individuals submit auto-generated code without transparency of how it was created. Engagement in collaborative work is strongly connected to trust in team members, and motivation to perform collaborative tasks may diminish when this is compromised (Dirks, 1999). Studies in software engineering emphasize the role of perceived transparency for trust (P. T. Y. Lee et al., 2024) as well as the role of perceived task-related competency of another team member (Mayer et al., 1995). Presumably, when team members use LLMs to generate code that in its form resembles more advanced programmers' code, their competency is much harder to judge, especially in the earlier stages of a project and by novices. As team members progress in collaborative tasks, building on others' code is necessary and requires judgment of the quality of that code. Studies about the relationship of LLM to the judgment of competency show that LLM use can lead novice programmers to misaligned confidence regarding their skills and understanding (Prather et al., 2024). Research has not yet addressed if the difficulty associated with the judgement of competency also applies to group-related judgement.

The difficulty in judging contributions might also affect the connectedness of the group. Previously, social connectedness, defined via measurements of frequency of social contact, task assistance and compassion, as well as sense of belonging, has been shown to be related to well-being (Frieling, M., Peach, E. K., & Cording, J., 2018). Connectedness can be defined as an affective outcome of group processes developed directly from interactions, such as mutual support, but also impressions of others, from their contributions against the context of own work on the common project.

2.3. The Role of Process in Collaborative Programming

I have argued that LLMs used by individuals in collaborative programming may affect learning and socio-affective outcomes. A sole focus on outcomes in a collaborative learning scenario is insufficient. Dillenbourg et al. (1996) argued that process variables must also be considered when studying collaboration. This is because interaction effects between the many process-related mediators of collaboration outcomes would prevent reliable causal inference. Given the dearth of research on process variables related to LLM-mediated contributions in a collaborative process, a relationship between the indicators of the process data with learners' perceptions of the members and the group need to be established.

For this, individual and group-level team code submissions need to be transformed to appropriate interaction process indicators. Log data from programming projects is different from conversation data often applied in collaboration research, though conceptual similarities exist. For example, students who work on programming projects regularly merge their modified versions of the software into a common version. How the students amend the versions and who does this gives insight into success of previous coordination as well as (perceived) value of the individual members' contributions and who maintains overview of the group's code. In some groups, major conflicts result from not being able to amend different versions to a working product (Tushev et al., 2018). In sum, group-level patterns of logs can make an impression on student perceptions of others and the group itself.

Moreover, depending on the type of contributions, individual roles in relation to the group may also be visible in GitHub traces. For example, previous work has talked about the phenomenon of "cowboy programming", where a group member took over the management of the relevant parts of the software development without including others (Tushev et al., 2018). Similarly, "free riders" and "social loafers" describe common patterns of individual roles students take on, bringing out negative group work dynamics (Nguyen et al., 2023).

Existing research on process indicators in programming creates a foundation for analyzing both individual and group processes. According to code collaboration project research, best- performing teams show equal contributions, not necessarily the highest total number of commits, but parallel main work times and work on separate branches of the code (Tushev et al., 2018). Team roles can be visible as one person contributing documentation while another contributes the code (Tushev et al., 2018). (Gitinabard et al., 2020) look at teamwork features on GitHub projects and classify the student teams into three groups, collaborative, cooperative, or solo-submit, differentiating each contribution into types such as bug fix, documentation, test case, or implementation. They look at how many lines of code the members change, across how many different files, how much they delete, and informativeness of commit messages written by a contributor. Another line of work goes deeper into contribution quality from code analysis and considers which contributions fix or keep problems in the code as indicated by build logs (Chen et al., 2022). These process measures have been analyzed in relation to student performance, but not in relation to student perceptions of each other and the group, as well as with LLM-ingestions within the contributions.

3. Research Questions

To investigate the effects of individual use of LLM-based tools on group-based learning in programming projects in higher education, this project poses the following research questions:

1. How does the use of LLM in collaborative programming affect individual outcomes, such as learning gain, self-efficacy, and professional identification, and group-related socio- affective attitudes, such as perceptions of teammate competency and connectedness?

2. What is the relationship between process indicators describing individual and group- level team code with the perceptions of competency among team members and group-related socio-affective attitudes?

4. Research Design

To address these research questions, a series of authentic studies in a university scenario are planned. I plan to collect the data from group-based programming projects from the courses targeting novice programmers. This choice is due to the focus on the first experiences with programming group projects within higher education. The courses are expected to give us a participant number of between 600 and 1800 students in 120 to 450 groups.

We will collect data via interviews, self- reports with established tools three times throughout the course and log data of the evolving code and all GitHub activities. The interviews are employed to potentially show causalities between our researched variables. Including a self- and team assessment part in group projects is a common strategy for their grading, and we will extend the usual questions with additional questions. A final presentation of the artifacts is part of the project's examination and questions about the team's code show each students' understanding of the software.

The first study will address the question of how the use of LLM in collaborative programming affects individual outcomes, such as learning gain, self-efficacy, and professional identification, and group-related socio-affective attitudes, such as perceptions of teammate competency and connectedness, also considering perceived equality of contributions. We will use self-reports for established measurement tools and interviews to capture these.

In the second study, we will analyze process data to evaluate relations to perceived effort distribution, accuracy of judgment in team members' competency, trust, differences in patterns in code contributions, mutual support, prevalent issues in engineering the software, perceived gain of skill, perceived appreciation from others and identification with the field of study and future professions.

Since I am at an early stage of this project, I am not set on the methods of analysis yet. We will tentatively analyze the first data and develop process measures appropriate to the theory of group dynamics in programming projects and relate them to self-report measures. The process measures will be sourced from log data of AI interactions, submitted milestone planning documents for coordination traces and features of the frequently logged contributions (using analytics of and building on previous code contribution evaluation metrics). The intra-group interactions will include information of who built on top of or modified whose code, who fixed code cohesion or formatted the others' code, who contributed comments or documentation, and where were errors introduced and resolved.

Acknowledgements

I would like to thank my supervisor, Professor Sasha Poquet, for her extraordinary support in my research work and learning, also in the advice given for this project.

References

[1] Bassner, P., Frankford, E., & Krusche, S. (2024). Iris: An AI-Driven Virtual Tutor For Computer Science Education. https://doi.org/10.1145/3649217.3653543

[2] Begel, A., & Nagappan, N. (2009). Coordination in Large-Scale Software Development: Helpful and Unhelpful Behaviors. https://www.microsoft.com/en-us/research/publication/coordination-in-large-scale-software-development-helpful-and-unhelpful-behaviors/

[3] Boyatzis, R., Rochford, K., & Cavanagh, K. V. (2017). Emotional Intelligence Competencies in Engineer's Effectiveness and Engagement. Career Development International, 22(1), 70–86. https://doi.org/10.1108/CDI-08-2016-0136

[4] Chang, R., & Fang, Z. (2022). Effect of AI- Assisted Tutoring on Student Self-Efficacy and Achievement. Educational Technology & Society, 25(4), 245-256.

[5] Chen, H.-M., Nguyen, B., & Dow, C.-R. (2022). Code-quality evaluation scheme for assessment of student contributions to programming projects. Journal of Systems and Software, 188, 111273. https://doi.org/10.1016/j.jss.2022.111273

[6] Darvishi, A., Khosravi, H., Sadiq, S., Gašević, D., & Siemens, G. (2024). Impact of AI Assistance on Student Agency. Computers & Education, 210. 104967. https://doi.org/10.1016/j.compedu.2023.104967

[7] Dillenbourg, P. (1990). What do you mean by "collaborative learning"? In Collaborative Learning: Cognitive and Computational Approaches (pp. 1-19). Pergamon.

[8] Dillenbourg, P., Baker, M., Blaye, A., & O'Malley, C. (1996). The Evolution of Research on Collaborative Learning. In Learning in Humans and Machine: Towards an Interdisciplinary Learning Science (pp. 189– 211). Elsevier.

[9] Dirks, K. T. (1999). The effects of interpersonal trust on work group performance. Journal of Applied Psychology, 84(3), 445–455. https://doi.org/10.1037/0021-9010.84.3.445

[10] Duhigg, C. (2019). What Google Learned From Its Quest to Build the Perfect Team. The New York Times.

[11] Finnie-Ansley, J., Denny, P., Becker, B. A., Luxton-Reilly, A., & Prather, J. (2022). The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming. Australasian Computing Education Conference, 10–19. https://doi.org/10.1145/3511861.3511863

[12] Frieling, M., Peach, E. K., & Cording, J. (2018). The Measurement of Social Connectedness and Its Relationship to Wellbeing. Ministry of Social Development New Zealand.

[13] Gitinabard, N., Estrada, T., & Bowers, A. (2020). Teamwork Features and Their Impact on Student Collaboration in GitHub Projects. Journal of Learning Analytics, 7(3), 25–45. https://doi.org/10.18608/jla.2020.73.3

[14] Hall, W. M., Schmader, T., & Croft, E. (2015). Engineering Exchanges: Daily Social Identity Threat Predicts Burnout Among Female Engineers. https://journals.sagepub.com/doi/abs/10.1177/1948550615572637

[15] Järvelä, S., & Hadwin, A. F. (2013). New Frontiers: Regulating Learning in CSCL. Educational Psychologist, 48(1), 25–39. https://doi.org/10.1080/00461520.2012.748006

[16] Kasneci, E., Seßler, K., Kuechemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh,

G., Günnemann, S., Hüllermeier, E., et al. (2023). ChatGPT for Good? On Opportunities and Challenges of Large Language Models for Education. Learning and Individual Differences, 103, 102274. https://doi.org/10.1016/j.lindif.2023.102274

[17] Kazemitabaar, M., Hou, X., Henley, A., Ericson, B. J., Weintrop, D., & Grossman, T. (2023). How Novices Use LLM-Based Code Generators to Solve CS1 Coding Tasks in a Self-Paced Learning Environment. arXiv. http://arxiv.org/abs/2309.14049

[18] Kazemitabaar, M., Ye, R., Wang, X., Henley, A. Z., Denny, P., Craig, M., & Grossman, T. (2024). CodeAid: Evaluating a Classroom Deployment of an LLM-based Programming Assistant that Balances Student and Educator Needs. Proceedings of the CHI Conference on Human Factors in Computing Systems, 1–20. https://doi.org/10.1145/3613904.3642773

[19] Khakurel, J., & Blomqvist, K. (2022). Artificial Intelligence Augmenting Human Teams. A Systematic Literature Review on the Opportunities and Concerns. In H. Degen & S. Ntoa (Eds.), Artificial Intelligence in HCI (pp. 51–68). Springer International Publishing. https://doi.org/10.1007/978-3-031-05643-7_4

[20] Lee, Y.-F., Hwang, G.-J., & Chen, P.-Y. (2022). Impacts of an AI-based chatbot on college students' after-class review, academic performance, self-efficacy, learning attitude, and motivation. Educational Technology Research and Development, 70(5), 1843–1865. https://doi.org/10.1007/s11423-022-10142-8

[21] Mayer, R. C., Davis, J. H., & Schoorman, F. D. (1995). An Integrative Model of Organizational Trust. The Academy of Management Review, 20(3), 709–734. https://doi.org/10.2307/258792

[22] Marra, R. M., Rodgers, K. A., Shen, D., & Bogue, B. (2009). Women Engineering Students and Self-Efficacy: A Multi-Year, Multi-Institution Study of Women Engineering Student Self-Efficacy. Journal of Engineering Education, 98(1), 27–38. https://doi.org/10.1002/j.2168- 9830.2009.tb01003.x

[23] Miller-Young, J., Jamieson, M., & Beck, S. (2023). Diverse Experiences and Belonging in an Online, First-Year, Team-Based Engineering Design Course. Teaching in Higher Education, 1–17. https://doi.org/10.1080/13562517.2022.2162816

[24] Nam, D., Macvean, A., Hellendoorn, V., Vasilescu, B., & Myers, B. (2024). Using an LLM to Help With Code Understanding. Proceedings of the IEEE/ACM 46th International Conference on Software Engineering, 1–13. https://doi.org/10.1145/3597503.3639187

[25] Prather, J., Reeves, B., Leinonen, J., Macneil, S., Randrianasolo, A., Becker, B., Kimmel, B., Wright, J., & Briggs, B. (2024). The Widening Gap: The Benefits and Harms of Generative AI for Novice Programmers.

[26] Salomon, G. (2003). Distributed Cognitions: Psychological and Educational Considerations. Cambridge University Press.

[27] Tushev, A., Gross, S., & Schaarschmidt, M. (2018). Collaborative Programming Dynamics in Software Engineering Education: Insights from Student Project Logs. ACM Transactions on Computing Education, 18(3), Article 14. https://doi.org/10.1145/3134822

[28] Vaithilingam, P., Zhang, T., & Glassman, E. L. (2022). Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models. CHI Conference on Human Factors in Computing Systems Extended Abstracts, 1–7. https://doi.org/10.1145/3491101.3519665

[29] Vygotsky, L. S. (1978). Mind in Society: Development of Higher Psychological Processes. Harvard University Press.

[30] Wang, L., Gao, R., Váncza, J., Krüger, J., Wang, X. V., Makris, S., & Chryssolouris, G. (2019). Symbiotic human-robot collaborative assembly. CIRP Annals, 68(2), 701–726. https://doi.org/10.1016/j.cirp.2019.05.002

[31] Wang, Q., Camacho, I., Jing, S., & Goel, A. K. (2022). Understanding the Design Space of AI-Mediated Social Interaction in Online Learning: Challenges and Opportunities. Proceedings of the ACM on Human-Computer Interaction, 6(CSCW1), 1–26. https://doi.org/10.1145/3512977

[32] Wang, Shaofeng, Zhuo Sun, & Ying Chen. (2023). Effects of Higher Education Institutes' Artificial Intelligence Capability on Students' Self-Efficacy, Creativity, and Learning Performance. Education and Information Technologies, 28(5), 4919–4939. https://doi.org/10.1007/s10639-022-11246-8