Use of Code Generation Models in Programming Education: a Systematic Literature Review

Doga Cambaz
d.cambaz@student.tudelft.nl

Supervisor: Xiaoling Zhang
Responsible Professor: Fenia Aivaloglou

1 Background

AI-driven code generation models offer opportunities and challenges for both **educators** and **students** in programming education, such as:

+ they allow students to concentrate on problem-solving components by correcting syntax errors [1], novice programmers perform better & faster with less frustration [2]
+ they help educators create curricula by generating programming exercises and solution explanations[1].
− there are concerns regarding academic integrity and users' over-reliance on auto-generated code [2].

These models have the potential to transform how programming is taught and learned. However, there is still **a lack of understanding of how best to adapt our educational practices to effectively manage the challenges and benefits** associated with their use.

2 The Research Questions

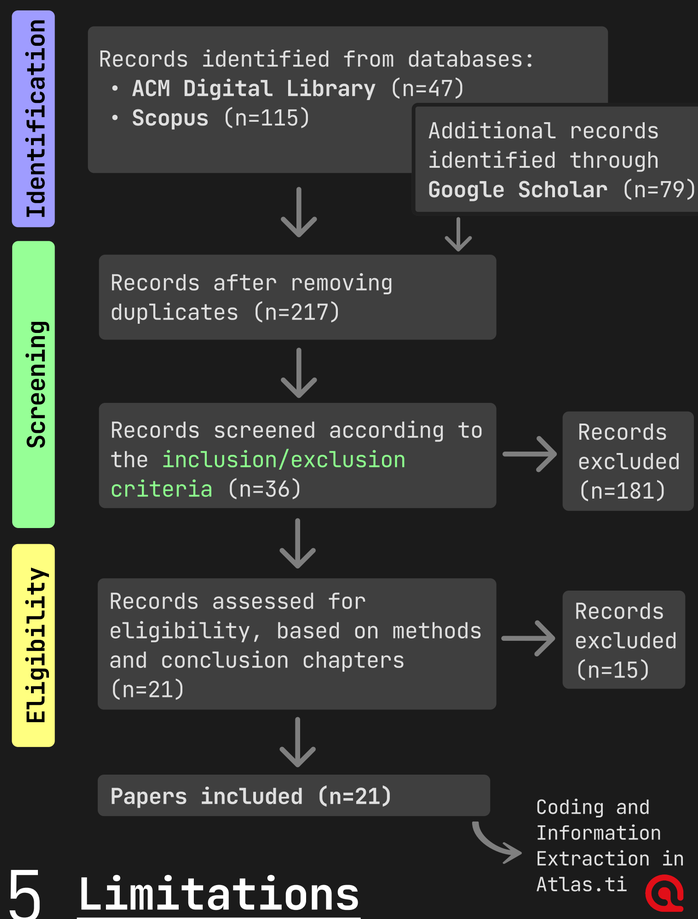
How can code generation models be used in practices for teaching and learning programming?

- **RQ1:** What are the practices that use code generation models for teaching and learning programming?
- **RQ2:** What are the characteristics of the code generation models that are used in teaching and learning practices?
- **RQ3:** What indicators are used for evaluating the performance of code generation models in teaching and learning practices?
- **RQ4:** What aspects should be considered when utilizing code generation models in teaching and learning practices?

3 Methodology

Systematic Literature Review
- to identify all empirical evidence that fits the pre-specified criteria and minimize research bias

PRISMA Flow Diagram: the paper selection process



4 Results

Teaching Programming: Assistive tools for assessment generation and evaluation

- Automatic generation of programming assignments, sample answers and explanation, test cases, variations of questions
- Grade Assignments, generate feedback, identify areas students are struggling

Learning Programming: Virtual Tutors for learners

- Generate practise exercises, examplar solutions and solution alternatives
- Explain and improve student code, clarify error messages and provide suggestions, support conceptual understanding, provide syntax tips

Evaluating the Code Generation Models

- evaluating generated content according to **qualitative criteria** (sensibleness, novelty, topicality, readiness for use) and **quantitative criteria** (accuracy, how many tests passed, how many lines of code is explained, etc.) [3,4]

Challenges and Ethical Considerations

- **Academic Integrity**
- **Over-reliance** on the tools, leading to **loss of creativity and critical thinking**
- **Appropriateness to novice programmers**
- **Accuracy and Reliability issues** of the models
- **Harmful biases in AI**
- **Code reuse and licensing issues**

What to do?

- embrace and integrate the code generation tools instead of focusing on detecting and preventing their use
- leave a transitional period for novice programmers

5 Limitations

- lack of solid guidelines
- the dynamic nature of large language models
- studies only focus on programming education in English

6 Conclusion

- Code generators in programming education presents a promising avenue with possibilities to improve student's learning experience and alleviate the workload of teachers.
- However, ensuring safe usage is crucial as failure to address the models' accuracy limitations, risk of misconduct and over-reliance pose a significant danger to computing education.
- Future studies should explore integrating AI code generators in classrooms and designing programming assessments that encourage critical thinking rather than relying on these tools as answer generators.

References

1. Brett A Becker et al. "Programming Is Hard - Or at Least It Used to Be: EducationalOpportunities and Challenges of AI Code Generation". In: Association for Computing Ma-chinery, 2023, pp. 500–506. isbn: 9781450394314. doi: 10 . 1145 / 3545945 . 3569759. url:https://doi.org/10.1145/3545945.3569759.
2. Majeed Kazemitabaar et al. "Studying the Effect of AI Code Generators on SupportingNovice Learners in Introductory Programming". In: Association for Computing Machinery,2023. isbn: 9781450394215. doi: 10.1145/3544548.3580919. url: https://doi.org/10.1145/3544548.3580919.
3. Paul Denny, Sami Sarsa, Arto Hellas, and JuhoLeinonen. Robosourcing Educational Resources –Leveraging Large Language Models for Learnersourcing. Nov. 2022. arXiv: 2211.04715 [cs.HC].
4. Sami Sarsa, Paul Denny, Arto Hellas, and JuhoLeinonen. "Automatic Generation of ProgrammingExercises and Code Explanations Using Large Lan-guage Models". In: Proceedings of the 2022 ACMConference on International Computing EducationResearch - Volume 1. ICER '22. Lugano and VirtualEvent, Switzerland: Association for Computing Ma-chinery, 2022, pp. 27–43. ISBN: 9781450391948. DOI:10.1145/3501385.3543957.