Software engineers still need **solid foundations**, despite the power of **LLMs**.

An Empirical Study on Usage and Perceptions of LLMs in a Software Engineering Project

Background: We conducted this study on a semester long, academic software engineering project (CS3203) with >200 students.

Methods

We encouraged students to utilize LLMs for code generation and introduced procedures for students to document the **prompts used**, **code integrated**, and the amount of **additional modification** required to integrate the output from the LLMs into the project. These information were documented in students' project files in the form of comments, which were extracted and analyzed at various milestones of the project timeline. We measured the amount of additional modification performed on the LLMs' output in three different levels of **human intervention**: none, minor (<10% of modified lines), and major (≥10%).

Problem statement: How would software engineers use LLMs in their project development, and would using LLMs significantly impact code quality and correctness? In addition, how much effort would be saved by using LLMs, if any?

Findings & Discussion

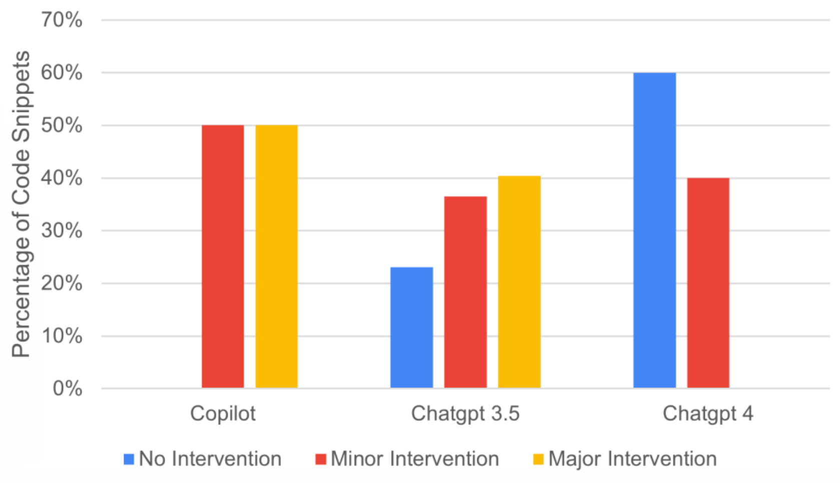
LLMs were used more **frequently** in earlier milestones and generated more **complex** output in later ones.

Milestone	Snippets	Lines	Lines / Snippet
MS1	53	1781	33.60
MS2	3	159	53.00
MS3	7	513	73.29
MS1 (aggregate)	53	1781	33.60
MS2 (aggregate)	56	1940	34.64
MS3 (aggregate)	63	2453	38.94

Prompts used fall into these 5 categories:

(1) **Improving code** included in the prompt.
(2) Simple tasks pertaining to the **language**.
(3) **Data** structures for specific requirements.
(4) **Algorithms** for common problems.
(5) Others (test cases, design patterns).

All LLMs used produced outputs that required **additional modification** before the code can be integrated into the project. There is still a need to **critically assess and integrate** AI-generated code, requiring solid foundations in software engineering.



We found **no significant difference** in the code **quality** and **correctness** between teams that had utilized LLMs heavily in the project, and teams that had not used LLMs to generate code. Students **more skilled** in coding were also **more inclined to use LLMs** for code generation, which is likely due to the additional skill requirement needed to understand the quality and correctness of the output and perform the necessary **modifications before integrating** the code into the project codebase.

LLM4Code at {ICSE'24}

Sanka Rasnayaka, Guanlin Wang,
Ridwan Shariffdeen, Ganesh Neelakanta Iyer

NUS National University of Singapore