AUTHOR Roelof van der Geest

Using local LLMs in constrained environments for increasing mutation score

1. BACKGROUND

- MUTATION TESTING

  Creating mutants of the existing source code to create intentional mistakes. Use these mutants to see if the existing tests for the program catch these mistakes.

- TEST GENERATION

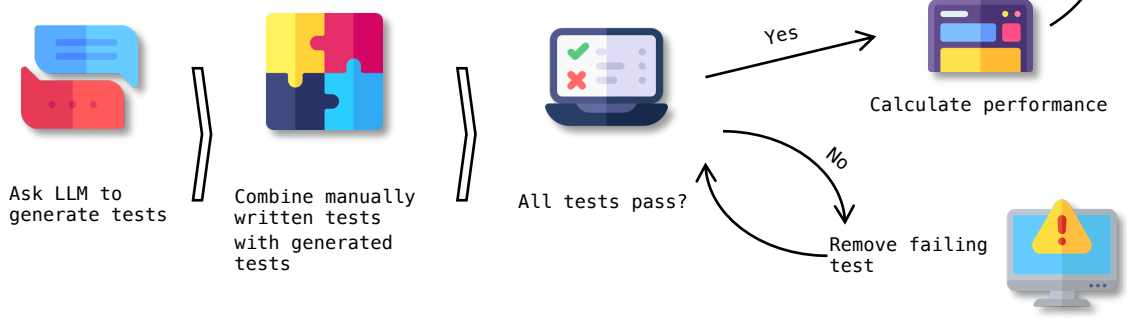
  SBST and fuzzing makes use of algorithms to search for new tests cases. For example tools like Evosuite and Randoop. These tests capture the program well, but are often not usable for various reasons.

2. RESEARCH QUESTION

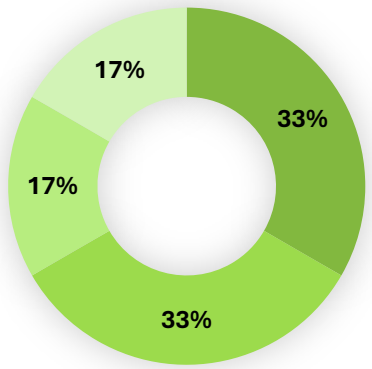
How effective are local LLMs running in constrained environments at increasing mutation score?

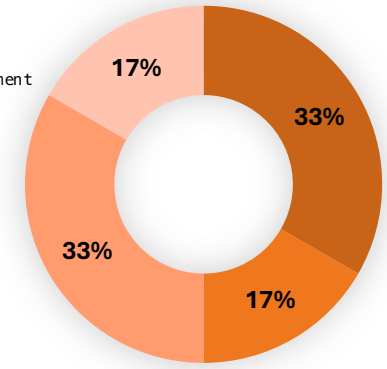
We compare 3 different LLM models (Deepseek Coder 6.7B, Code Llama 13B and Codestral 22B), to see how effective they are at increasing mutation score by generating unit tests for Java.

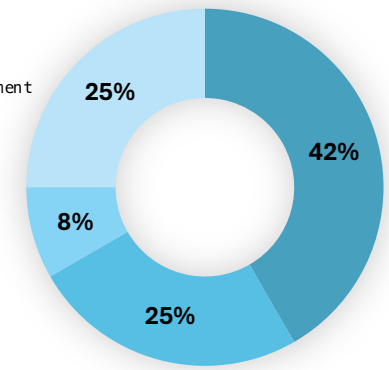
3. METHOD



4. RESULTS







Model	Runtime
Deepseek Coder 6.7B	15-25s
Code Llama 13B	25-100s
Codestral 22B	20-35s

5. DISCUSSION

- CODE FROM TRAINING SET PERFORMS BETTER

  Some classes in the corpus are available on GitHub. All three models were more effective at increasing mutation score for these classes, as opposed to classes that were not on GitHub, and thus likely not in the training set.

- COMMENTS ARE IMPORTANT

  Adding comments to code impacts the ability of a model to generate tests. Classes with comments score higher than classes without comments.

- MODEL SIZE IS LESS SIGNIFICANT

  The performance difference between models with 6.7B, 13B and 22B parameters is not very large. All models were able to generate new tests at about the same efficiency. The runtime is also comparable.

- LANGUAGE IS EXTREMELY IMPORTANT

  LLMs appear to look at more than just syntactic and semantic meaning of code. Classes written in a foreign language throw off all three models so much, they are not able to generate any passing tests.

6. LIMITATIONS

- PROMPT MIGHT NOT BE OPTIMAL

  Even though prompt engineering is outside the scope, we performed preliminary testing to find a prompt which yields good results.

- CLASSES MIGHT BE PART OF TRAINING SET

  To minimize this risk, the majority of classes was taken from the SF110 dataset, which is likely not part of the training set

- CLASSES MIGHT NOT BE REPRESENTATIVE

  We selected classes with a cyclomatic complexity of at least 10, and from a variety of different projects with different purposes

7. CONCLUSION

The proposed approach is effective at increasing the mutation score of manually written tests. The mutation score can be increased in 4 to 6 out of 12 cases, depending on the model. The most important factors that impact performance are the language of the code (English only) and the presence of comments.

Future research can focus on 1) prompt engineering, which could improve results even more, and 2) new models appear at a rapid rate, which could also bring performance improvements.

SUPERVISORS A. Panichella, M. Olsthoorn