Using LLM-Generated Summarizations to Improve the Understandability of Generated Unit Tests

Natanael Djajadi <n.djajadi@tudelft.nl>

1. Introduction

Background

- Generated tests can be **hard** to understand
- Factors that play a role in understandability [1]:
  - Test names, comments, **test summarizations,** and more

Limitations of generated summaries [2]

- Can be **lengthy** and **redundant**
- Best to use **in combination** with well-defined test names and variables

Existing tools

- `TestDescriber` [3]: template-based approach to generate summaries
- `DeepTC-Enhancer` [2]: template-based approach to generate summaries and deep-learning to rename variables
- **UTGen** [4]: Evosuite + Large Language Models (LLMs) to increase understandability

Research gap

- Extend UTGen with LLM-generated summaries

2. Main Research Question

To what extent can the **understandability** of a test case be influenced by Large Language Model-generated test summaries in terms of **context**, **conciseness**, and **naturalness**?

3. Methodology

Phase I – Experimenting

Prompting Techniques

- Simple prompt – *Baseline*
- Prompt engineering – *UTGen template + Chain-of-Thought*
- Few-shot approach – *Include code demonstrations*
- Context-awareness – *Include method under test*

Large Language Models

- `Codellama:7b-instruct` – 7 billion parameters
- `ChatGPT3.5` – 175 billion parameters
- `ChatGPT4o` – 1.76 trillion parameters

Methodology

- Run the 3 LLMs with the 4 techniques 3 times for 2 classes
- **Length** of output & **understandability**
  - Understandability: Judge with pre-defined rules

Phase II - Evaluation

- **User study/evaluation** with 11 participants
- 4 rounds with 4 different method summaries
- Find **characteristic elements**
  - Using pre-defined approach
  - Encourage participants to come up with their own

4. Results

RQ1 *Impact on understandability by using different prompt techniques*

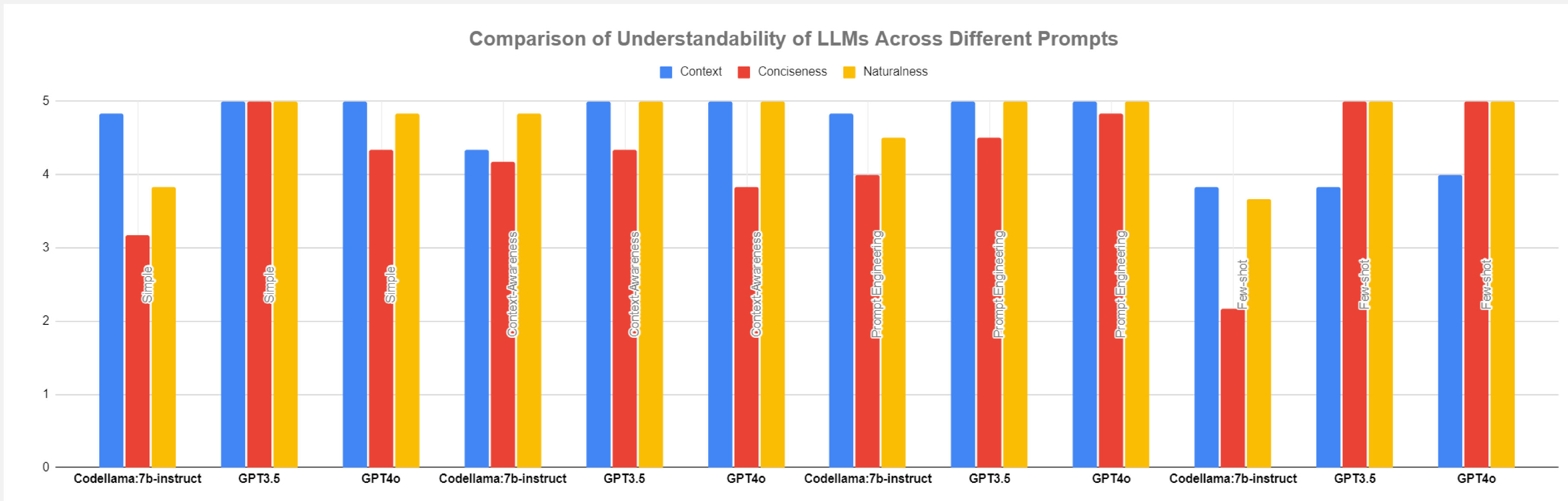

Figure 1: Results from the comparison for the summaries of the LLMs across different prompts

RQ2 *Comparative influence of LLM-generated summaries and existing tools*

Tool	Context	Conciseness	Naturalness
TestDescriber	3.97	2.85	3.21
DeepTC	3.88	4.21	3.64
Codellama:7b-instruct	4.48	3.93	4.09
ChatGPT	3.78	4.57	4.05

Table 1: Average results of the aspects (context, conciseness, naturalness) of understandability of the LLM-generated summaries

Tool	# times favored	Understandability
TestDescriber	7	3.34
DeepTC-Enhancer	12	3.09
ChatGPT	17	4.17
codellama:7b-instruct	20	4.13

Table 2: Comparison of the number of times participants preferred to use a summary generated by the tool and the average score of understandability

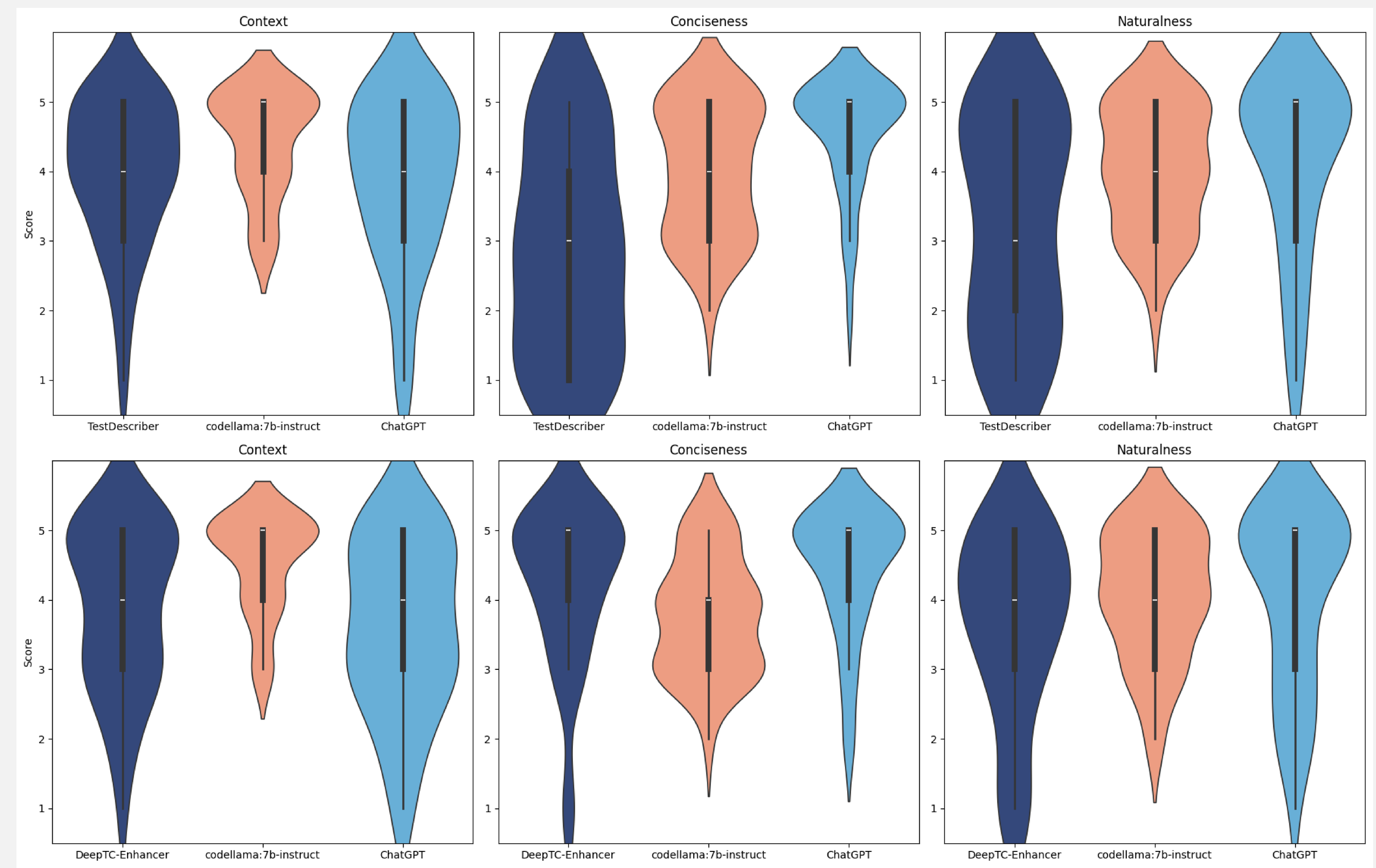

Figure 2: Results from the comparison for the summaries of the LLMs using a 5-point Likert scale

RQ3 *Differences in test summary elements influencing understandability*

`TestDescriber`

Pros
- ✓ Step-by-step guide on main aspects

Cons
- ✗ Inline comments
- ✗ Long
- ✗ Contains unnecessary information

`DeepTC-Enhancer`

Pros
- ✓ Good summarization of the flow of test
- ✓ Concise
- ✓ Numbered list (step-by-step guide)

Cons
- ✗ Line-by-line analysis

`Codellama:7b-instruct`
with prompt engineering

Pros
- ✓ Detailed and clear description of test
- ✓ Structured format

Cons
- ✗ Lengthy
- ✗ Line-by-line analysis

`ChatGPT`
with few-shot

Pros
- ✓ Concise
- ✓ Directly addresses test's objective

Cons
- ✗ Explaining too little

5. Conclusion & Future Research

Conclusion

- LLM-generated tools scored **higher** and were **favored** over existing tools
- Multiple aspects influence **understandability** with own advantages and disadvantages

Main Research Question Answer

- **Right** prompting technique + **suitable** LLM = **positive** impact on **understandability** ✓
  - *Context*: Prompt engineering (`codellama:7b-instruct`)
  - *Conciseness*: Few-shot with the right LLM (`ChatGPT`)
  - *Naturalness*: LLM-generated summaries

Insight

- Results indicate that participants **prefer** context over conciseness

Future research

- **User evaluation** for different prompting techniques of different LLMs
- Prompt engineered `ChatGPT` **vs** prompt engineered `codellama:7b-instruct`

References

[1] D. Winkler, P. Urbanke, and R. Ramler. "Investigating the readability of test code". In: Empirical Software Engineering 29.2 (Feb. 2024), p. 53.n ISSN: 1573-7616. DOI: 10.1007/s10664-023-10390-z.URL: https://doi.org/10.1007/s10664-023-10390-z.

[2] D. Roy et al. "DeepTC-enhancer: improving the readability of automatically generated tests". In: ASE '20: Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering, Jan. 2021, pp. 287–298. DOI: 0.1145/3324884.3416622. URL: https://dl.acm.org/doi/10.1145/3324884.3416622.

[3] S. Panichella et al. "The impact of test case summaries on bug fixing performance: an empirical investigation". In: ICSE '16: Proceedings of the 38th International Conference on Software Engineering, May 2016, pp. 547–558. DOI: 10.1145/2884781.2884847. URL: https://dl.acm.org/doi/10.1145/2884781.2884847.

[4] S. Panichella et al. "The impact of test case summaries on bug fixing performance: an empirical investigation". In: ICSE '16: Proceedings of the 38th International Conference on Software Engineering, May 2016, pp. 547–558. DOI: 10.1145/2884781.2884847. URL: https://dl.acm.org/doi/10.1145/2884781.2884847.

More information

Responsible Professor: Andy Zaidman <A.E.Zaidman@tudelft.nl>
Supervisor: Amirhossein Deljouyi <A.Deljouyi@tudelft.nl>