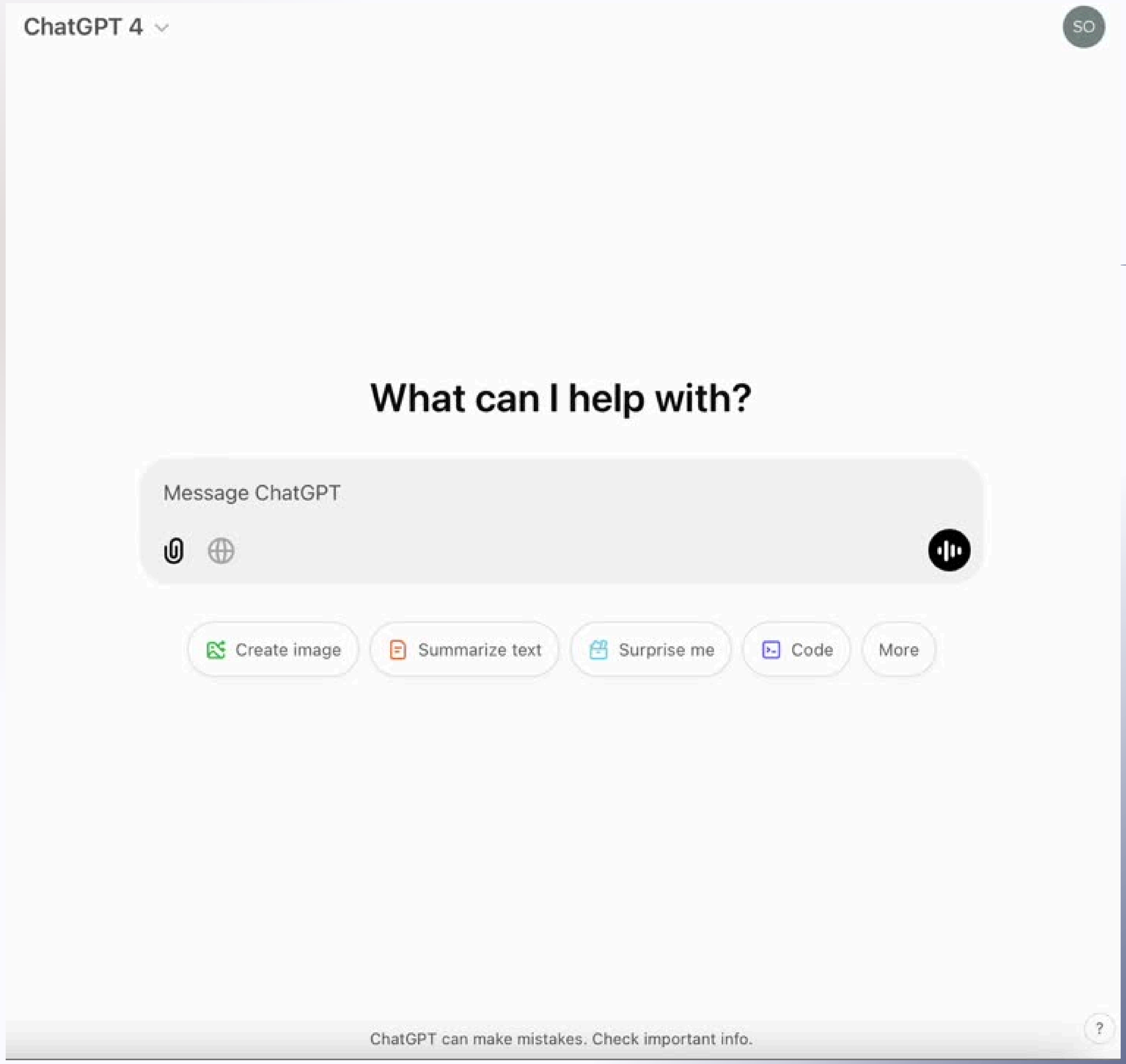
AUTOREPAIRABILITY OF CHATGPT AND GEMINI : A COMPARATIVE STUDY

Chutweeraya Sriwilailak*, Yoshiki Higo†, Pongpop Lapvikai*, Chaiyong Ragkhitwetsagul* and Morakot Choetkiertikul*
*Faculty of Information and Communication Technology (ICT), Mahidol University, Nakhon Pathom, Thailand
†Graduate School of Information Science and Technology, Osaka University, Osaka, Japan

INTRODUCTION

Large Language Models (LLMs) are improving at **coding tasks fixing program errors,** handling diverse issues more flexibly than traditional methods. Using **Autorepairability** [1] to measure success with LLMs, we identify key code functionality that impact results.



Reference: [1] P. Lapvikai, C. Ragkhitwetsagul, M. Choetkiertikul, and Y. Higo, “Autorepairability: A new software quality characteristic,” in SANER’24, 3 2024, pp. 787–791.

WHAT IS AUTO PROGRAM REPAIR TOOLS ?

- Automated program repair (APR) tools are used to **automatically identify and fix bugs** in software code.
- APR can help **improve programmer productivity and software quality**.
- Some techniques used in APR for search-based approaches , pattern-based repair , and constraint-based approaches .

RELATED WORK

AUTOREPAIRABILITY: A NEW SOFTWARE QUALITY CHARACTERISTIC [1]

Autorepairability is a quality measure that checks how well APR (Automated Program Repair) tools like **“kGenProg”**[2] can fix bugs in a project. It helps developers decide if a system is suitable for automatic repairs. By **measuring “autorepairability”**[1] score, developers can see how effective APR tools are for their projects, making software maintenance more efficient and reliable

[2]Y. Higo, S. Matsumoto, R. Arima, A. Tanikado, K. Naitou, J. Mat- sumoto, Y. Tomida, and S. Kusumoto, “kGenProg: A High-Performance, High-Extensibility and High-Portability APR System,” in APSEC’18, 2018, pp. 697–698.

STEP TO MEASURE AUTOREPAIRABILITY[1]

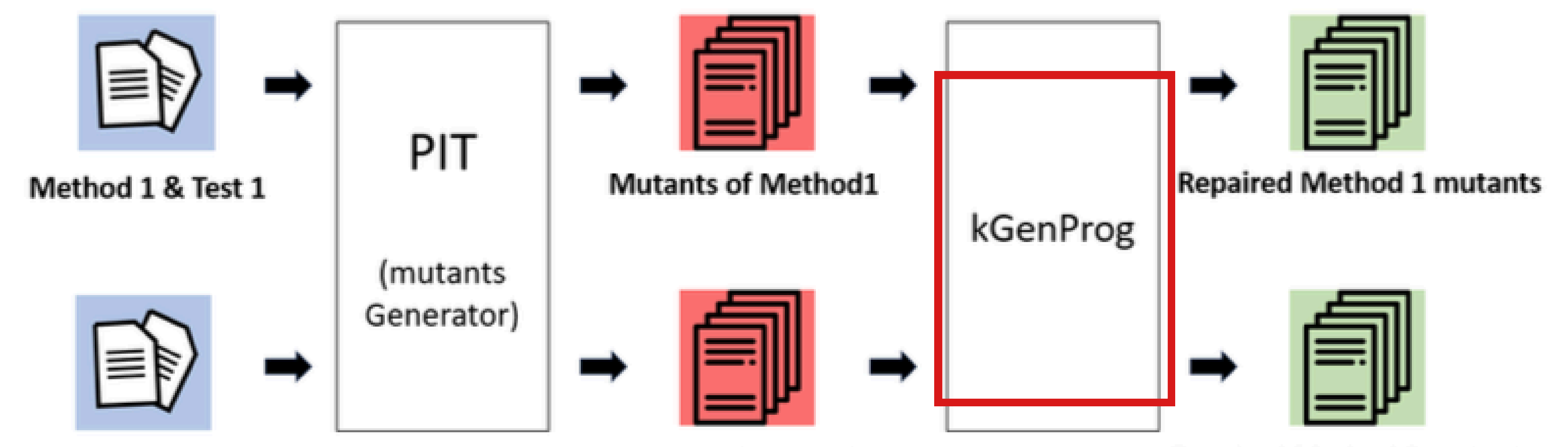


Fig. 1: Steps in our experiment

STEP 1: Generating Mutants

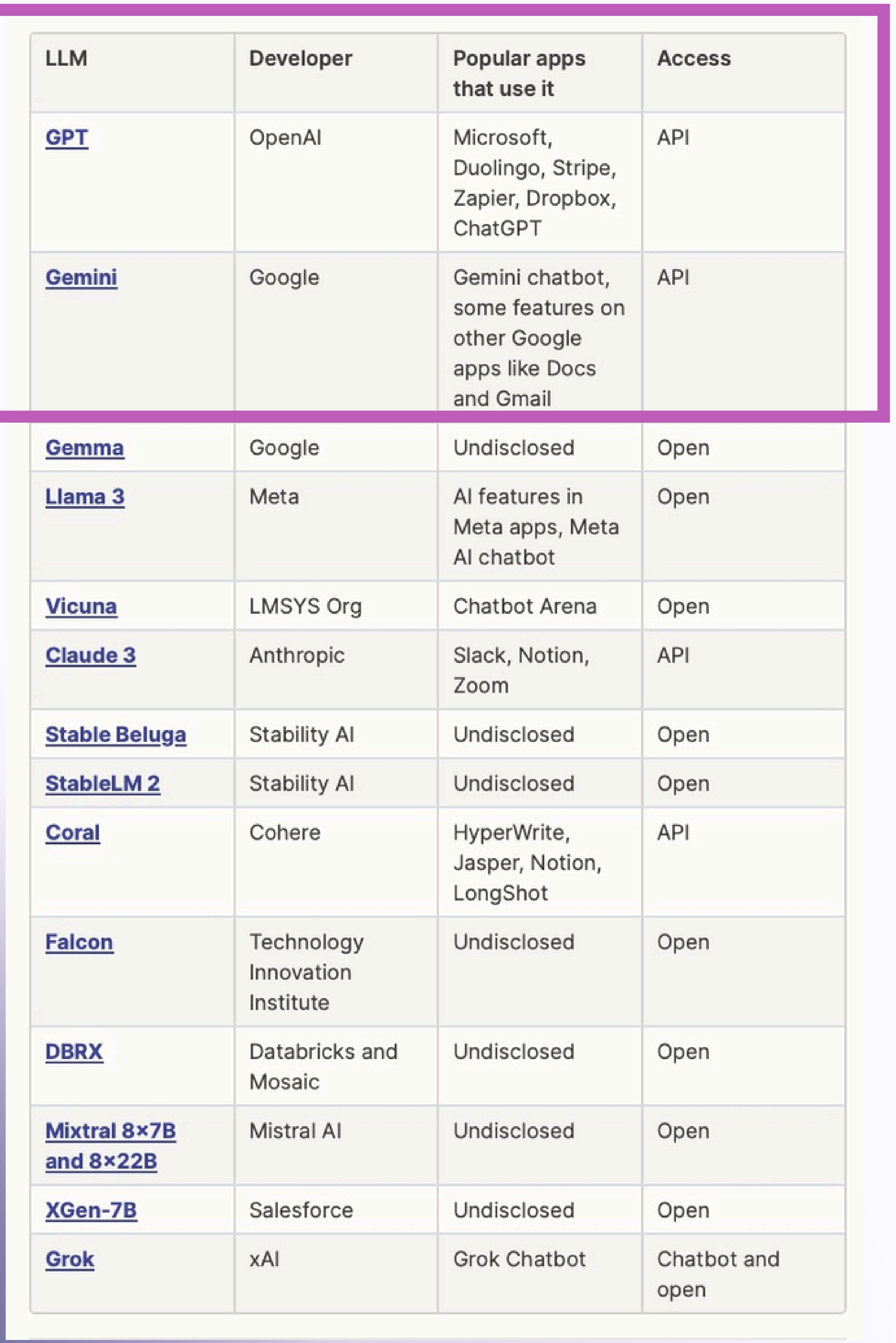
STEP2: Applying AUTO PROGRAM REPAIR tool

STEP3: Calculating AR-ability Score

Calculate Autorepairability Score: Compute the autorepairability score as the ratio of successful solutions I S I to the total mutants I M I

AUTOREPAIRABILITY = |S| / |M|

WE CAN REPLACE OTHER TOOLS INSTEAD OF KGENPROG IN ORDER TO CALCULATE AUTOREPAIRABILITY SCORE



LLM	Developer	Popular apps that use it	Access
GPT	OpenAI	Microsoft, Duolingo, Stripe, Zapier, Dropbox, ChatGPT	API
Gemini	Google	Gemini chatbot, some features on other Google apps like Docs and Gmail	API
Gemma	Google	Undisclosed	Open
Llama 3	Meta	AI features in Meta apps, Meta AI chatbot	Open
Vicuna	LMSYS Org	Chatbot Arena	Open
Claude 3	Anthropic	Slack, Notion, Zoom	API
Stable Beluga	Stability AI	Undisclosed	Open
StableLM 2	Stability AI	Undisclosed	Open
Coral	Cohere	HyperWrite, Jasper, Notion, LongShot	API
Falcon	Technology Innovation Institute	Undisclosed	Open
DBRX	Databricks and Mosaic	Undisclosed	Open
Mixtral 8x7B and 8x22B	Mistral AI	Undisclosed	Open
XGen-7B	Salesforce	Undisclosed	Open
Grok	xAI	Grok Chatbot	Chatbot and open

Reference: https://zapier.com/blog/best-llm/

SELECTED LLMS FOR EXPERIMENT

GPT 3.5 VS GEMINI 1.5

```
DATA
├──── Pair1
│    ├──── Pair1_Method1
│    │    ├──── 1
│    │    │    └──── Target.java
│    │    ├──── 2
│    │    ├──── 3
│    │    ├──── 4
│    │    └──── test
│    │         ├──── Target_ESTest_scaffolding.java
│    │         └──── Target_ESTest.java
│    ├──── log.csv
│    └──── Pair1_Method2
├──── Pair2
├──── Pair3
└──── Pair4
...
└──── Pair 1342
```

DATASET[1]

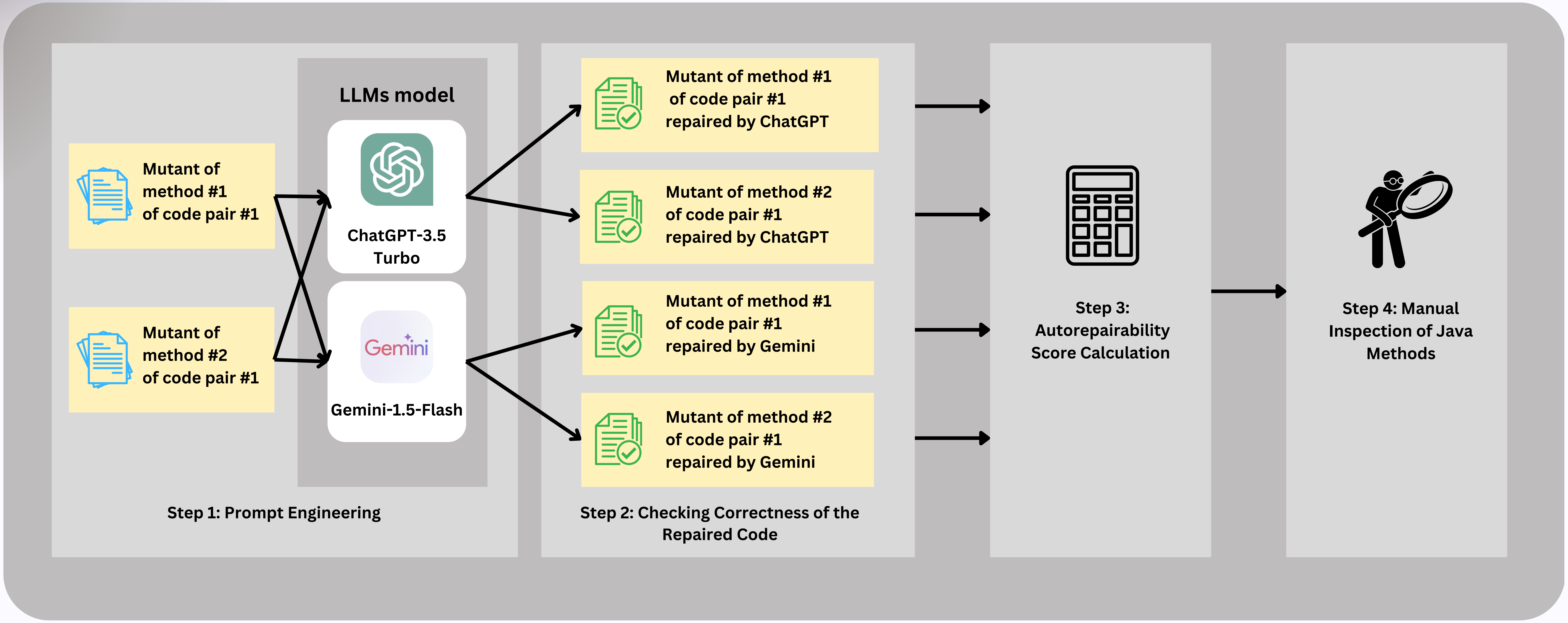
RESEARCH QUESTIONS

RQ1: What are the Autorepairability scores of ChatGPT and Gemini?

RQ2: What are the functionalities that affect the Autorepairability of the two LLMs?

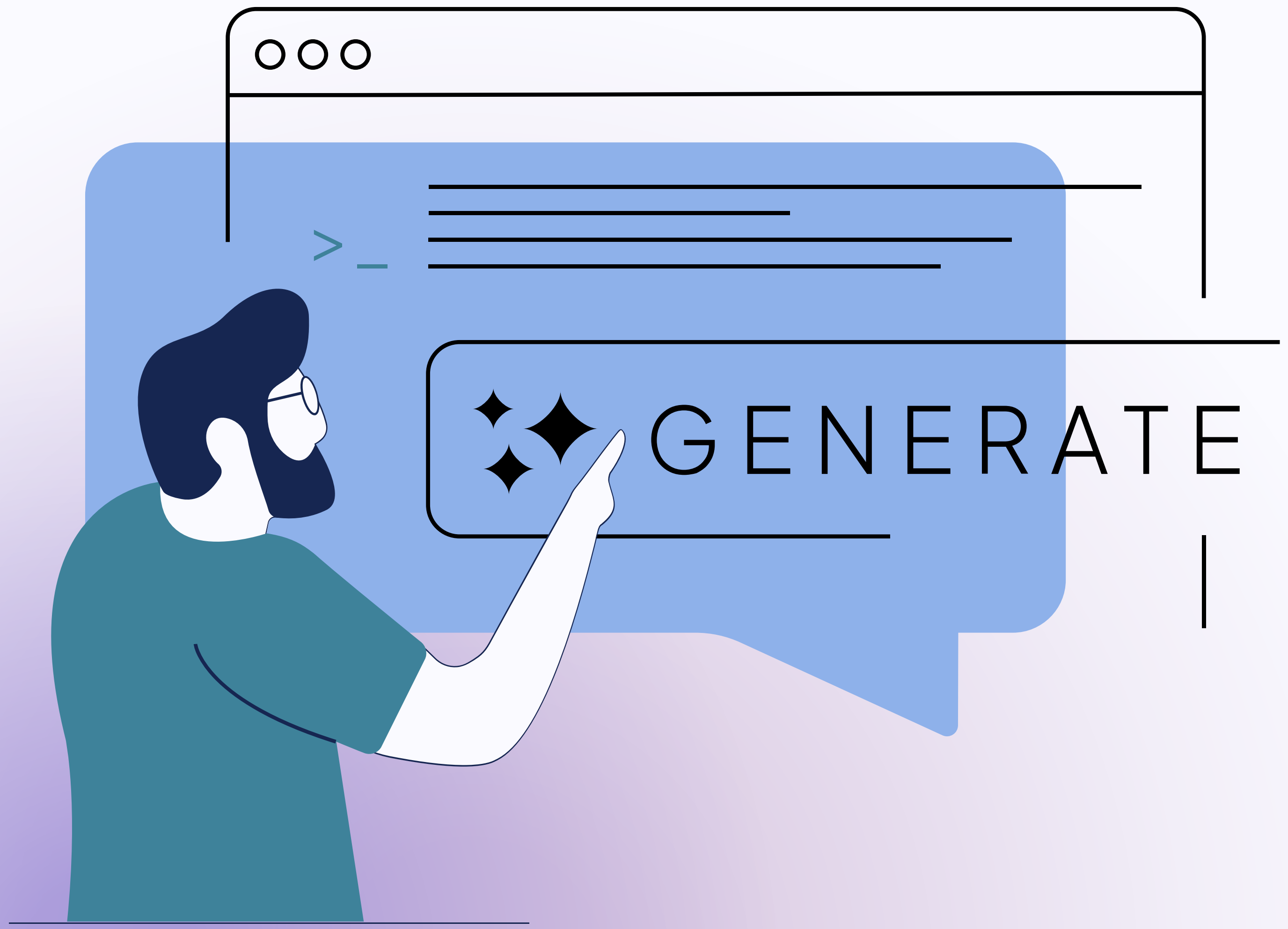
METHODOLOGY

METHODOLOGY


Mutant of method #1 of code pair #1
Mutant of method #2 of code pair #1
LLMs model
ChatGPT-3.5 Turbo
Gemini
Gemini-1.5-Flash
Step 1: Prompt Engineering
Mutant of method #1 of code pair #1 repaired by ChatGPT
Mutant of method #2 of code pair #1 repaired by ChatGPT
Mutant of method #1 of code pair #1 repaired by Gemini
Mutant of method #2 of code pair #1 repaired by Gemini
Step 2: Checking Correctness of the Repaired Code
Step 3: Autorepairability Score Calculation
Step 4: Manual Inspection of Java Methods

STEP 1: PROMPT ENGINEERING

The study evaluates two models using a consistent prompt template with Java code and test cases. Models fixed the code to pass all tests, returning only the corrected code for analysis.



Example of prompt that we use for sending in both ChatGPT and Gemini

~~~
[Code of the Java method to be repaired]
[Unit test cases of the method]
From the Java code above, this code fail on some test case.
Please update the code to make it run pass all the test case.
Respond only with the updated Java code (do not include
the test code) in this format:
```java
Repaired code
```
~~~

Pair Method	Result	total_tests_run	total_failures	Model
Pair1_Method1_01	Success	5	0	gemini-1.5-flash
Pair1_Method1_02	Success	5	0	gemini-1.5-flash
Pair1_Method1_03	Success	5	0	gemini-1.5-flash
Pair1_Method1_04	Success	5	0	gemini-1.5-flash
Pair1_Method2_01	Failed	5	1	gemini-1.5-flash

STEP 2: CHECKING CORRECTNESS OF THE REPAIRED CODE

The repaired code from the models was tested against the test cases that exposed the bugs to evaluate the fixes. Passing all test cases indicated '**Success,**' while failures were marked as '**failed**'.

STEP 3: AUTOREPAIRABILITY SCORE CALCULATION

AUTOREPAIRABILITY = NO. OF SUCESS / NO. TOTAL MUTANT

method_base	total_mutants	success	failed	autorepairability
Pair1_Method1	4	4	0	1
Pair1_Method2	4	2	2	0.5
Pair2_Method1	1	1	0	1
Pair2_Method2	1	1	0	1

RESULT ANALYSIS

RQ1: WHAT ARE THE AUTOREPAIRABILITY SCORES OF CHATGPT AND GEMINI?

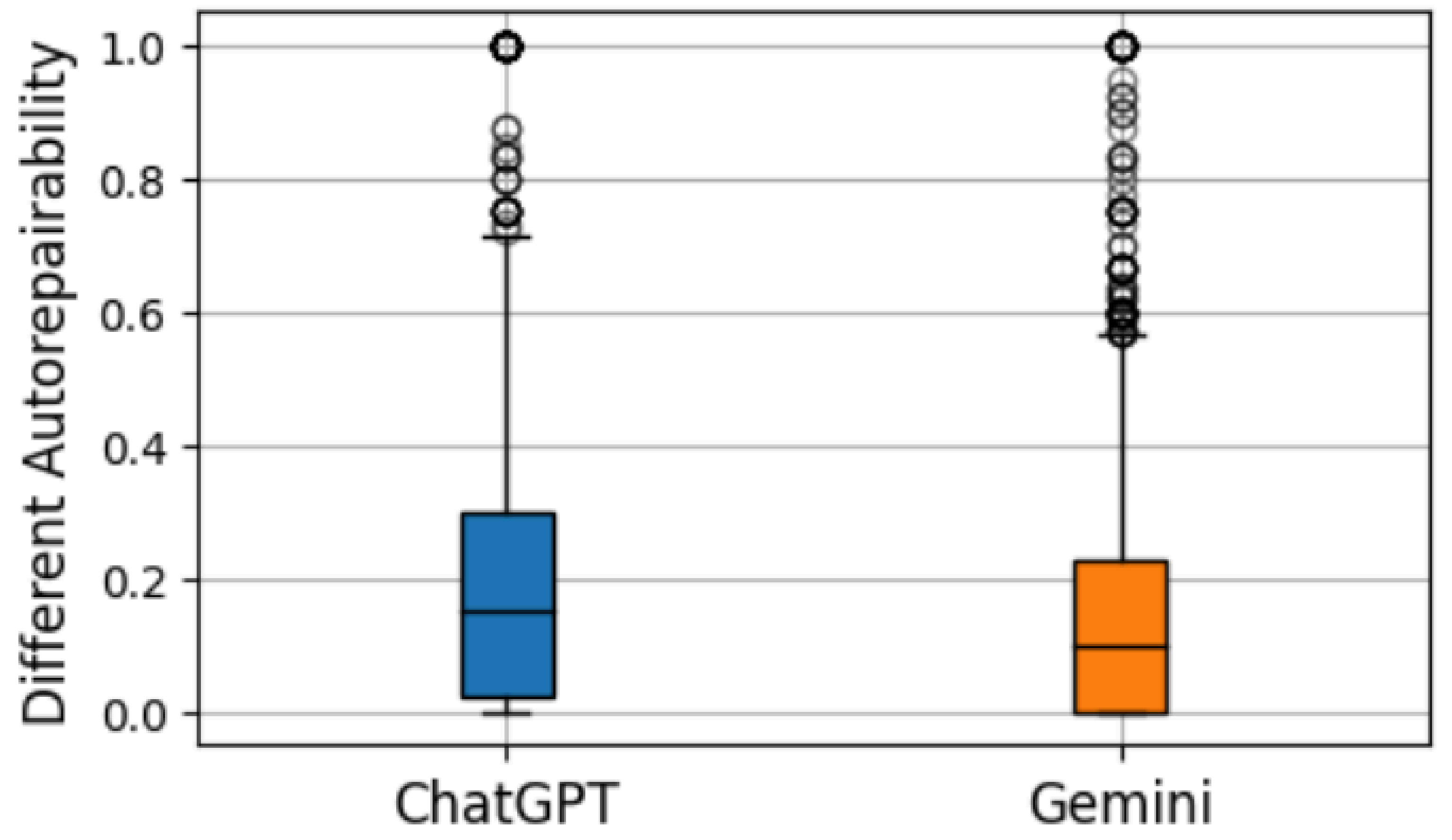
COMPARISON OF AUTOREPAIRABILITY

value	ChatGPT	Gemini
Max	1.00	1.00
Min	0.00	0.00
Median	0.43	0.77
Standard Deviation	0.30	0.31
Average	0.44	0.69

Differences of Autorepairability Scores of Method Pairs

The study analyzed autorepairability scores of 1,282 method pairs with similar functionality but differing structures, with Gemini achieving a median score of 0.10 compared to ChatGPT's 0.15.

method_base	total_mutants	success	failed	autorepair ability	Differences between pair
Pair1_Method1	4	4	0	1	0.5
Pair1_Method2	4	2	2	0.5	
Pair2_Method1	1	1	0	1	0
Pair2_Method2	1	1	0	1	



COMPARE AUTOREPAIR SCORE ACROSS MODEL

method_base	GPT_autorepairability	Gemini_autorepairability	GPT-GEMINI	GEMINI-GPT
Pair1_Method1	0.00	1.0	-1.00	1.00
Pair1_Method2	0.25	0.5	-0.25	0.25
Pair2_Method1	1.00	1.0	0.00	0.00
Pair2_Method2	1.00	1.0	0.00	0.00
Pair3_Method1	0.00	0.0	0.00	0.00
Pair3_Method2	0.00	0.0	1.00	0.00

CERTERIA

- the difference >0.5
- the total mutant > 10

GEMINI>GPT

130

GPT>GEMINI

2

RQ2: What are the functionalities that affect the Autorepairability of the two LLMs?

Key differences in Autorepairability were linked to five common coding functionalities:

1. **Geographic and Mathematical Operations:**
   Precision tasks like mapping and spatial analysis.
2. **Validation, Comparison, and Searching**
   Ensuring data integrity through validation and comparisons.
3. **Data Conversion**
   Transforming data formats, such as bytes to integers.
4. **Data Extraction and Comparison**
   Extracting and comparing key data elements.
5. **Encoding Operations**
   Handling data encoding for numerical or bit sequences.

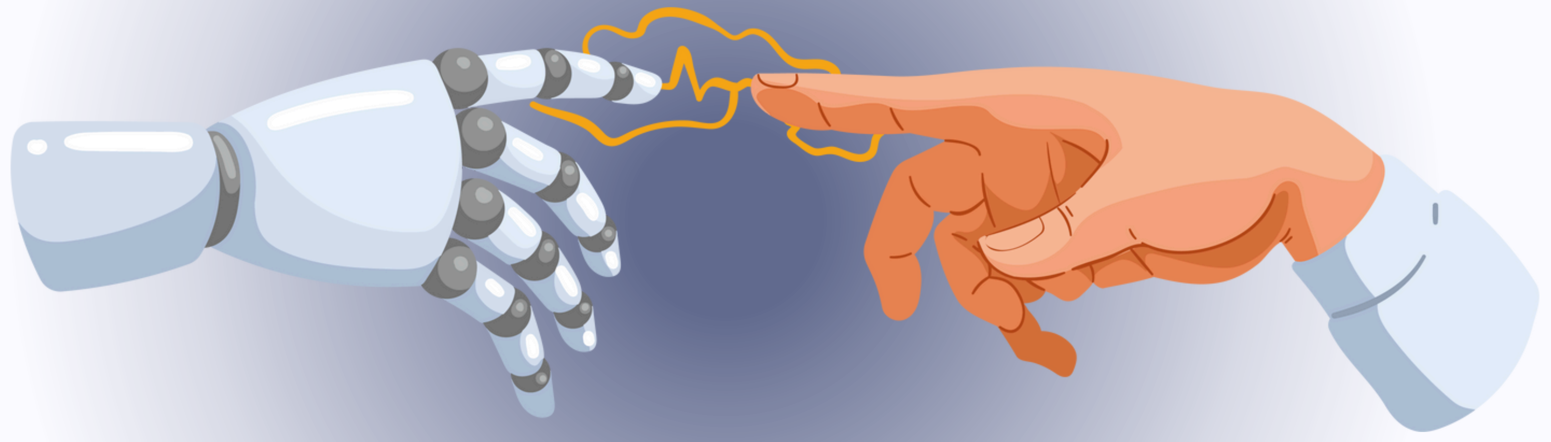
THREATS TO VALIDITY

Internal Validity

- Single investigator risks error and bias.
- Prompts significantly influence outcomes.

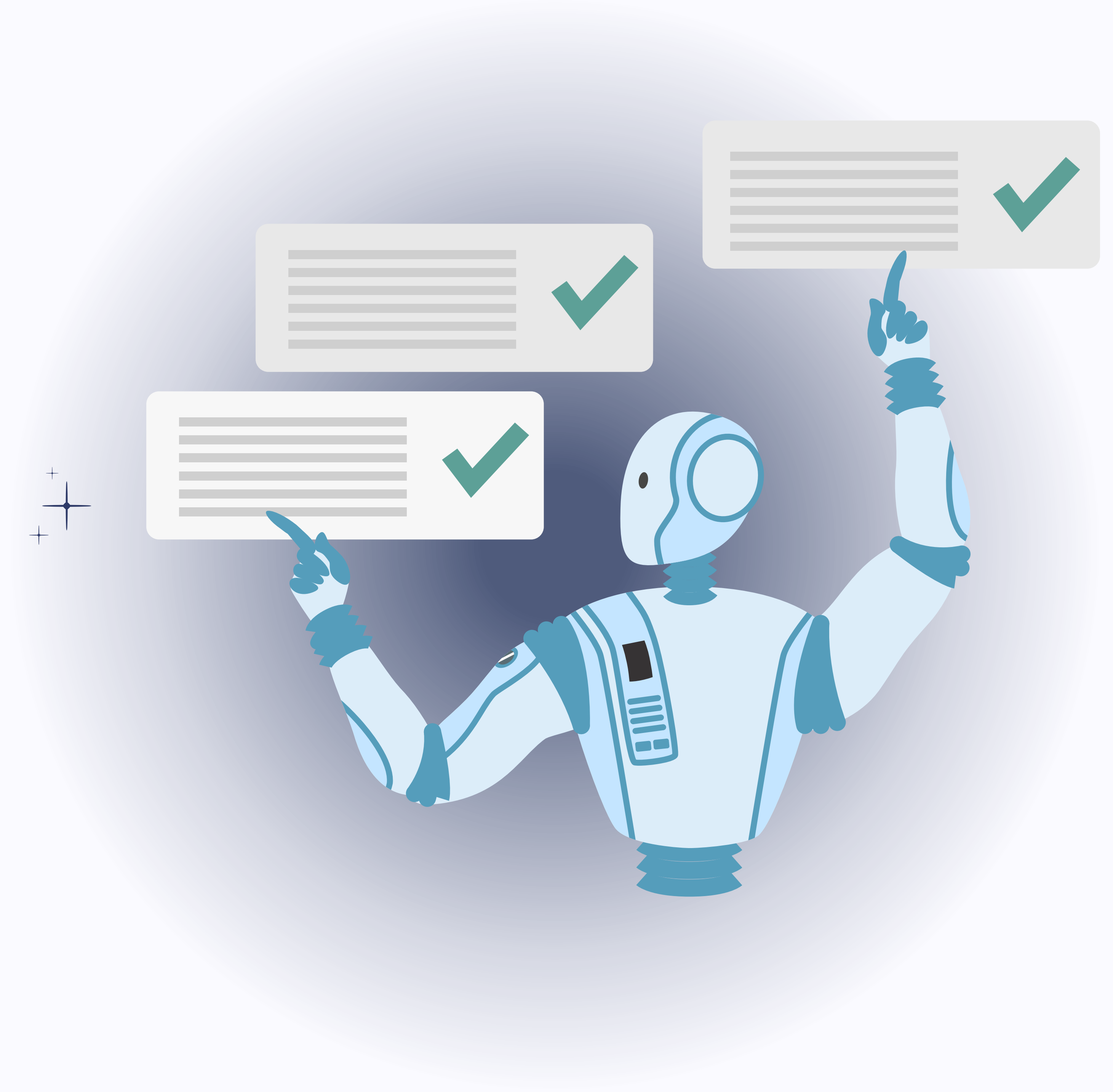
External Validity

- Findings limited to one dataset.
- Results vary across LLM versions.



SUMMARY

This study shows that **Gemini outperforms** ChatGPT in fixing bugs across 1,282 Java methods, particularly excelling in five key functionalities



FUTURE WORK

- Expand the study to include **more LLMs,** such as Llama, Claude, and other open-source models.
- **Repeat the experiments** with additional datasets, particularly real-world software projects, to enhance generalizability.

THANK YOU !