11 Sep 2024

LLM Agent to Automate Test Execution for Mobile Applications

Shu-Wei Cheng

Kamakshi Kodur

Yue Hu

Yao-Sheng Tsai

Zening Li

See next page for additional authors

Inventor(s)
Shu-Wei Cheng, Kamakshi Kodur, Yue Hu, Yao-Sheng Tsai, Zening Li, Fei Wang, Zhewen Song, and Oscar Ding

LLM Agent to Automate Test Execution for Mobile Applications

ABSTRACT

A substantial fraction of software or application development effort is spent on testing and debugging. For mobile device software, manual quality assurance (QA) testing on devices can be a time-consuming, repetitive, and error-prone task. This disclosure describes techniques that leverage a large language model (LLM) to automate execution of test cases, to perform visual assertions on screenshots, and to identify and report bugs in the software being tested. By leveraging the capabilities of LLMs to understand natural language instructions (e.g., input prompts), generate text (e.g., based on input test cases), perform image recognition tasks and answer visual questions (e.g., to provide visual assertions based on screenshots obtained from simulation), the techniques automate repetitive tasks/feature tests, improve the accuracy of QA testing, and reduce overall time required to perform tests, providing substantial cost savings and improved software quality.

KEYWORDS

- Large language model (LLM)
- LLM agent
- Quality assurance (QA)
- Software testing
- QA testing
- Mobile app testing
- Visual assertion
- Test case execution
- Mobile simulator

BACKGROUND

A substantial fraction of software or application development effort is spent on testing and debugging. The costs of testing and maintaining new versions of software can escalate substantially over time. For mobile device software, manual quality assurance (QA) testing on devices can be a time-consuming and repetitive task. Testers manually execute test cases, capture screenshots, and compare them to expected results to identify discrepancies. Manual QA testing can be tedious and error-prone, especially for complex applications with large numbers of test cases.

Large language models (LLMs) are artificial intelligence models capable of understanding natural language instructions, generating text, performing image recognition tasks, and answering visual questions.

DESCRIPTION

This disclosure describes techniques that leverage a large language model (LLM) to automate execution of test cases, to perform visual assertions on screenshots, and to identify and report bugs in the software being tested.

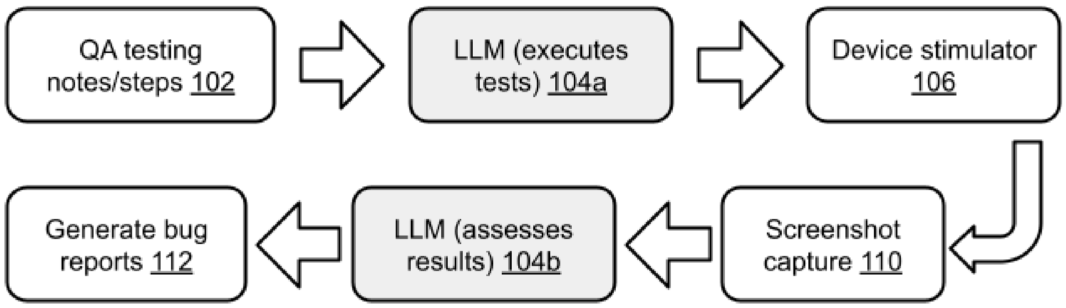
QA testing notes/steps 102
LLM (executes tests) 104a
Device stimulator 106
Screenshot capture 110
LLM (assesses results) 104b
Generate bug reports 112

Fig. 1: Use of a large language model (LLM) to test a mobile application

Fig. 1 illustrates the use of a large language model (LLM) to test a mobile application. To perform automated testing using an LLM, the following components are defined:

- *Test case input* (102) is a component that accepts input test cases in the form of quality assurance (QA) testing documents or testing steps.
- *Large language model* is a component that executes test cases (104a) and performs visual assertions on resulting screenshots (104b). The LLM can be provided the test case and a prompt that provides instructions regarding the test case execution task.
- *Mobile simulator* (106) is a component that simulates a mobile device on which the software under test (mobile app) is executed, and simulates user actions such as login, button-tapping, app navigation, etc.
- *Screenshot capture* (110) is a component that captures screenshots of the software under test (mobile app) as the simulator performs various steps of the test.
- *LLM-based assertion* (104b) is a component, which prompts the LLM to perform visual assertions on the screenshots to identify anomalies, e.g., deviation from expected results, changes from a previous version, etc.
- *Bug reporting* (112) is a component that automatically generates reports of observed anomalies, in a manner similar to the reports presently generated by manual QA.

Some advantages of using an LLM to automate test case execution and assertion regarding anomalies include:

- *Reduced maintenance overhead*: In contrast to traditional end-to-end tests, the described techniques obviate the substantial manual effort needed to write and to maintain feature tests. The LLM automates the test actions of manual QA testers, e.g., based on previously written test cases.
- *Reduced manual testing*: The use of an LLM to automate test execution and visual assertions can substantially reduce the overall cost and time required for testing.

- *Improved feature throughput*: The reduced test times enables improvements in the throughput and quality of end-to-end feature testing.
- *Improved product quality*: With the provision of appropriate prompts, an LLM can perform general visual assertions with high accuracy, which reduces the risk of false positives and false negatives.
- *Scalability*: The use of an LLM in test automation improves scalability, as a large number of test cases and mobile devices can be included in the test.
- *Flexibility*: It is easy to adapt the framework described herein to different testing scenarios and requirements. It is relatively easy to create, add, and iterate through general-purpose prompts that are adaptable to a variety of testing scenarios.

By leveraging the capabilities of LLMs to understand natural language instructions (e.g., input prompts), generate text (e.g., based on input test cases), perform image recognition tasks and answer visual questions (e.g., to provide visual assertions based on screenshots obtained from simulation), the techniques automate repetitive tasks/feature tests, improve the accuracy of QA testing, and reduce overall time required to perform tests, providing substantial cost savings and improved software quality.

CONCLUSION

This disclosure describes techniques that leverage a large language model (LLM) to automate execution of test cases, to perform visual assertions on screenshots, and to identify and report bugs in the software being tested. By leveraging the capabilities of LLMs to understand natural language instructions (e.g., input prompts), generate text (e.g., based on input test cases), perform image recognition tasks and answer visual questions (e.g., to provide visual assertions based on screenshots obtained from simulation), the techniques automate repetitive tasks/feature tests, improve the accuracy of QA testing, and reduce overall time required to perform tests, providing substantial cost savings and improved software quality.

REFERENCES

1. Wang, Fei; Kodur, Kamakshi; Micheletti, Michael; Cheng, Shu-Wei; Sadasivam, Yogalakshmi; Hu, Yue; and Li, Zening, "Large Language Model Driven Automated Software Application Testing", Technical Disclosure Commons, (March 26, 2024) https://www.tdcommons.org/dpubs_series/6815