EpicAutomator

Tucker Cook
Jacob Greenberg
Guillermo Rached

The Problem

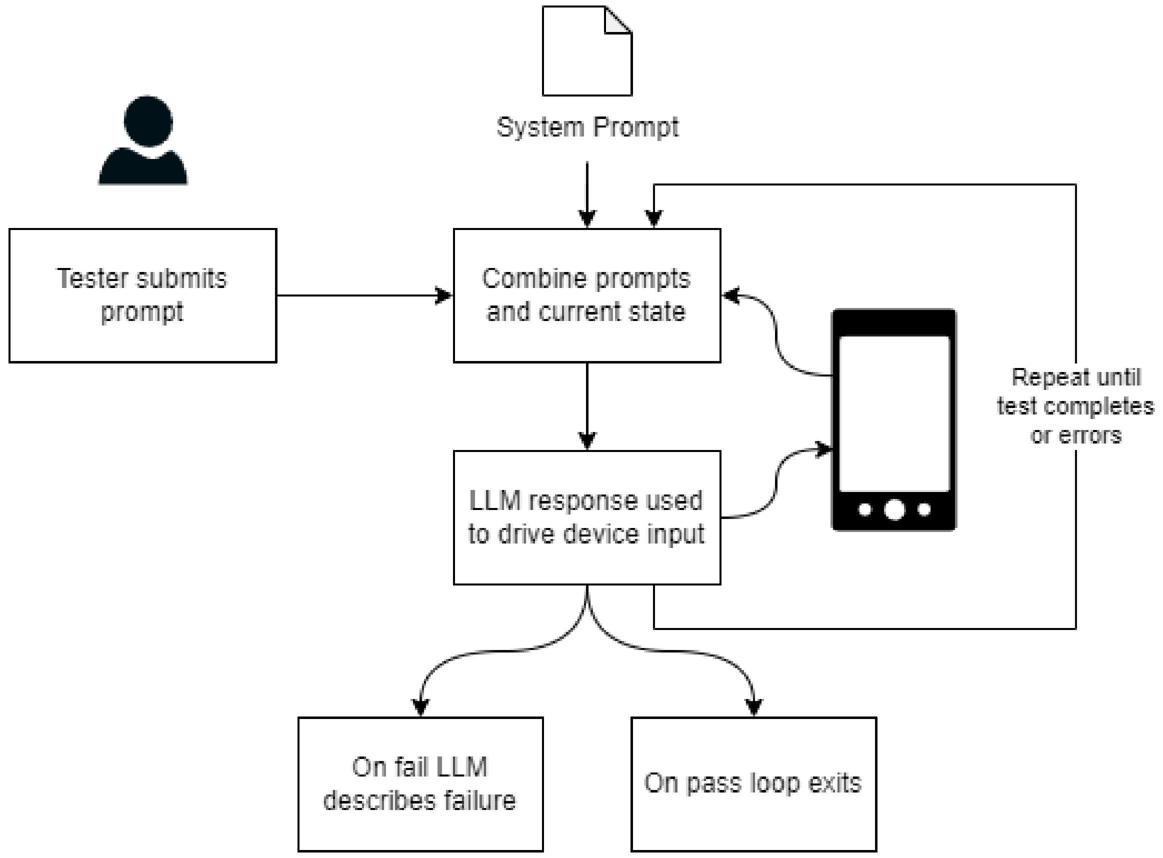
UI tests are obtuse and frustrating to automate.
Traditional solutions require:

- Updating the tests with every UI change
- Heavily sensitive to subtle initial state changes
- Poorly report error cases

Implementation

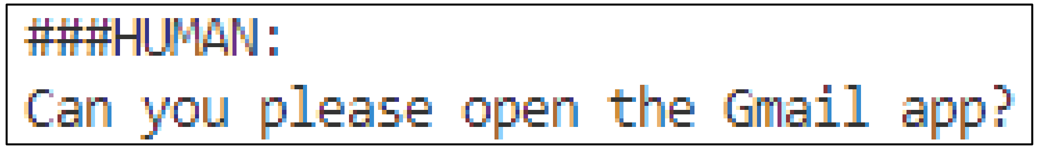
We sought to use an LLM to attempt to automate testing. Our system works by accepting a test prompt from a user. This is fed to an LLM which uses the prompt combined with screen data to drive device inputs.

We went to great lengths to keep the program modular and accessible so it can be easily iterated upon by users.


System Prompt
Tester submits prompt
Combine prompts and current state
Repeat until test completes or errors
LLM response used to drive device input
On fail LLM describes failure
On pass loop exits

Challenges

We found the locally hosted LLMs to be less cooperative than we expected. The LLM would occasionally ignore prompts to ask the human tester questions.

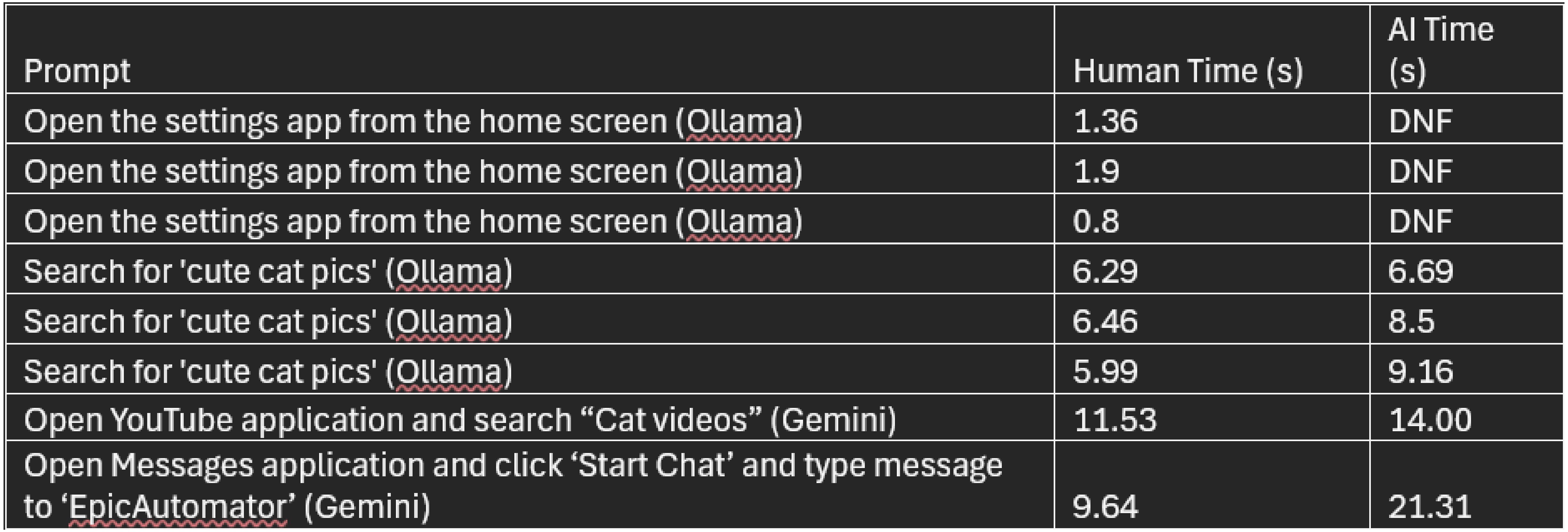


```
###HUMAN:
Can you please open the Gmail app?
```

Despite trying a number of models, the LLMs never seemed to grasp what they were supposed to do and seem to generate more or less random inputs.

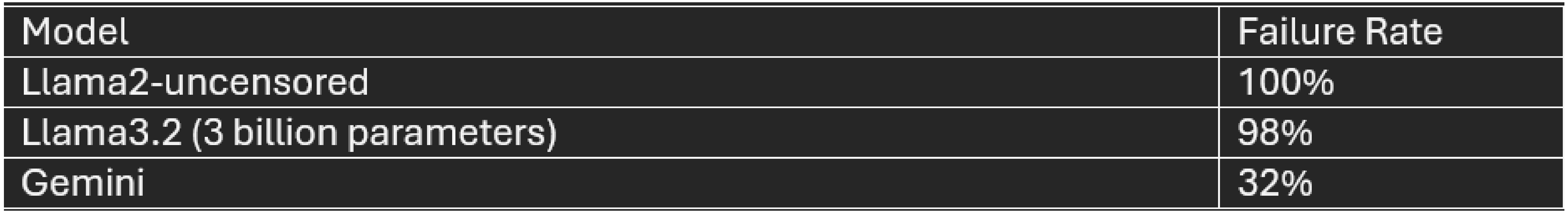
We ultimately decided to switch focus to remotely hosted LLMs like Gemini and began finding success with them.

Test framework



Prompt	Human Time (s)	AI Time (s)
Open the settings app from the home screen (Ollama)	1.36	DNF
Open the settings app from the home screen (Ollama)	1.9	DNF
Open the settings app from the home screen (Ollama)	0.8	DNF
Search for 'cute cat pics' (Ollama)	6.29	6.69
Search for 'cute cat pics' (Ollama)	6.46	8.5
Search for 'cute cat pics' (Ollama)	5.99	9.16
Open YouTube application and search “Cat videos” (Gemini)	11.53	14.00
Open Messages application and click ‘Start Chat’ and type message to ‘EpicAutomator’ (Gemini)	9.64	21.31

Test framework (cont.)

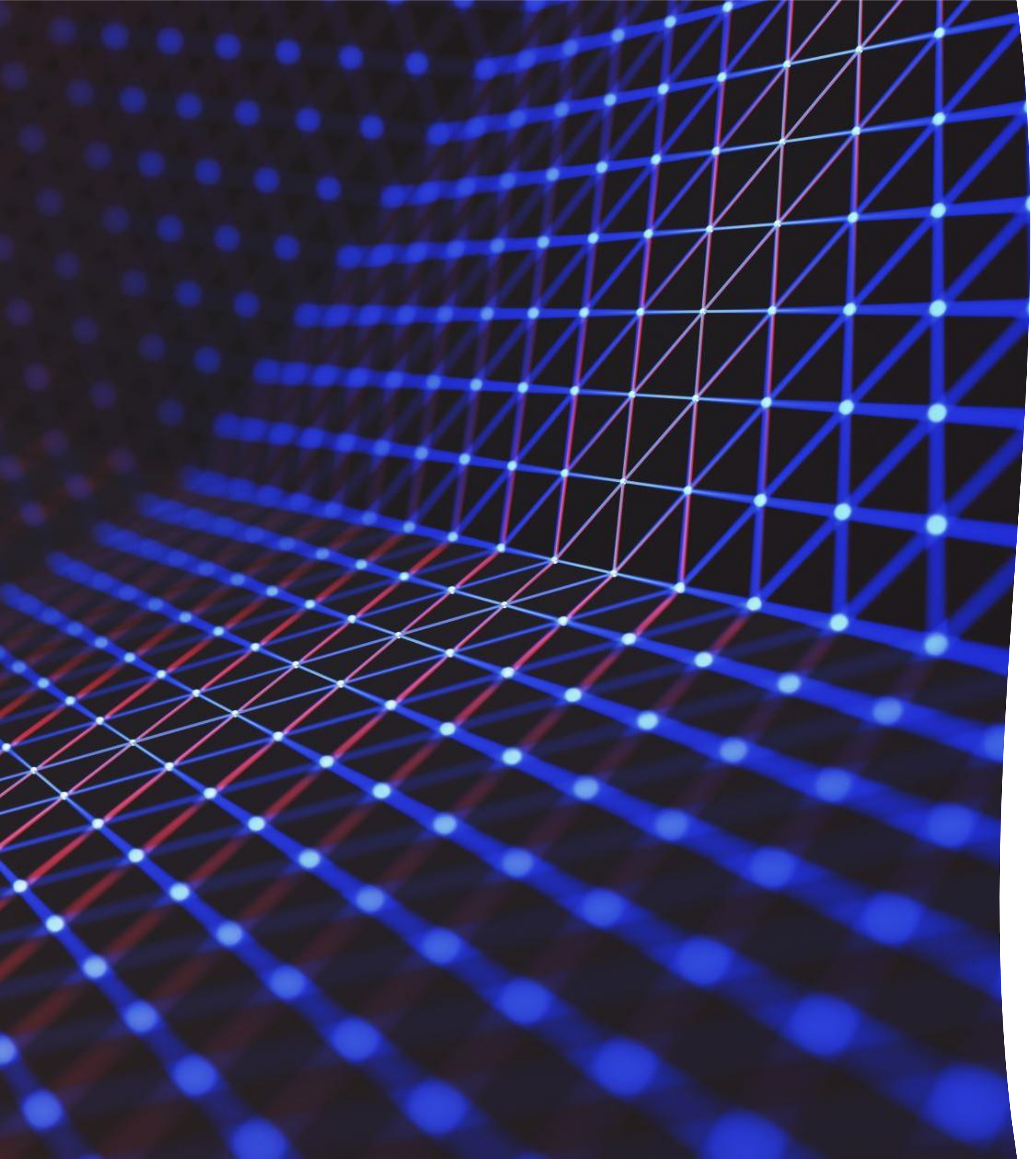


Model	Failure Rate
Llama2-uncensored	100%
Llama3.2 (3 billion parameters)	98%
Gemini	32%

The Results

With powerful LLMs we've shown that this is a feasible way to test UIs. State-of-the-art LLMs are capable of understanding both the XML and the prompt and using that information to navigate the device.

Unfortunately, private, locally hosted LLMs didn't fair so well. They clearly understood the prompt and made an effort to interact with the phone, but due to limitations with the context window usually weren't able to view the entire screen hierarchy.



Conclusion

We believe the introduction of LLMs into GUI based testing presents a promising future for automated testing.

We're very happy with the results of the project, especially when using Gemini as the LLM.

Future Work

Use of more advanced (paid) general purpose LLMs

Using an LLM with image recognition capabilities to enhance navigation

Using reinforcement learning to build an ML model better attuned to this kind of work.