WHITE PAPER

Artificial Intelligence (AI):

Using Image Compare with Computer Vision for Testing

Jonathan Stonesifer, Senior Software Test Analyst
Rajiv Ramanjani, Senior AI Architect
FIS Global Automation and Testing Services

November 2018

Computer Vision

This white paper examines how Artificial Intelligence (AI) tools such as Image Compare with Computer Vision can improve the accuracy and speed of User Interface (UI) testing that traditionally required manual attention. Technology has evolved to the point where AI tools can detect UI issues down to the pixel level. Utilizing this type of tool provides the opportunity to improve test results and keep pace with the time-sensitive and essential role of Quality Assurance (QA) testing in today's Agile software development world.

UI Testing Challenges and Opportunities

Challenge – We live in a new digital age where intelligent automated technology is replacing the repetitive manual processes that were performed in the past. This wave of innovation has touched every area of the banking business from internal operations to product solutions and will continue to expand into areas where data analysis and repetition are present. One area which has made unbounded leaps in automation is the field of Computer Vision. This technology uses a combination of Artificial Intelligence (AI) and Machine Learning (ML) to perform tasks previously performed through human analysis using sight. For example, law enforcement uses this technology for reading and processing license plates of cars traveling on public highways and byways. This same application of Computer Vision can also be used in the software testing industry as a replacement for smoke test verification in a Continuous Integration/Continuous Delivery (CI/CD) pipeline.

Traditionally smoke testing a User Interface (UI) has been a manual process, with manual intervention required to detect minute differences between UI screens that could otherwise be missed. With an increase in the number of screens being tested, the probability of manual testing errors increases. Also, certain aspects such as different fonts and colors (which could be different in different UI screens) add to the complexity and contribute to increased inefficiency.

When it comes to image elements testing, using the Functional Testing paradigm is not enough to verify whether an image appears and shows content as expected. Therefore, QA automation testers leave such elements for manual testing only. After some time, the manual image check can become burdensome, as it is highly repetitive work. The chance for bug leakage increases, as testers tend to check such features less often and with less detailed test cases.

Opportunity – The normal smoke testing process performed using automated test scripts can be quite time-consuming when executed properly. It essentially touches every page of an application, as well as every UI element related to the major functionality on those pages. When testing on a mid to large size application, the amount of data to be verified can become extremely large and take hours to finish.

The recent AI explosion has brought a new wave of technologies that stem from Big Data processing and cognitive machine learning. These technologies were developed in some of the most prestigious universities and academic learning institutions around the world and have always been intended as free-to-use software built upon existing open source architectures. These basic libraries are available to be utilized by anyone whom so wishes, and the door is now open for custom technology stacks with custom automation frameworks to integrate easily with these base libraries.

AI/ML represents a tremendous opportunity for the next generation of automated testing capabilities, and it will quickly become a necessity considering the leaps and bounds of advancements that technology continues to make.

Benefits

Benefits of AI/ML range from speed and efficiency to quality and consistency. The use of AI in a Computer Vision application will further enhance the accuracy of the function. High level benefits include:

- Elimination of the manual comparison of images, via the introduction an automated, quicker, and more accurate and efficient way of comparing images and UI screens.
- Capability to spot minute differences that might otherwise be missed by manual inspection, including identification of pixel level differences for fonts, colors etc.
- Solution delivery as an API service over the internet.
- Does not require extensive processing capabilities or any additional expensive software or hardware. Because the AI/ML solution resides on the internet as an API service, all processing happens on the internet rather than locally.

Use Cases of AI/ML Automated UI Testing Solutions in the Industry

Screenster

This is a commercially available solution where automated screen testing is provided as a service:

https://screenster.io/image-comparison-tool/

Eggplant

Eggplant Functional is a test automation solution that dynamically tests the UI. Instead of hooking into the code, the software tests from the user’s perspective through intelligent image and text understanding, API automation, and WebDriver object automation — all within a single test.

https://eggplant.io/

TestingWhiz

TestingWhiz is a web UI automation testing tool for validating the UI of web applications and the front end to confirm accurate appearance across different browsers, platforms and devices. TestingWhiz’s web test automation solution also helps testers authenticate critical functioning of applications to deliver an efficient and effective user experience.

By automating UI testing using TestingWhiz, testers can easily capture two web interfaces, compare the results with the expected output and record pixel by pixel differences based on a custom-defined tolerance level. TestingWhiz can also automate web UI testing across two URLs, two image files, or between an image and a URL.

https://www.testing-whiz.com/

Technologies and Architecture Used

OpenCV-Python

Core package used to highlight the differences between images

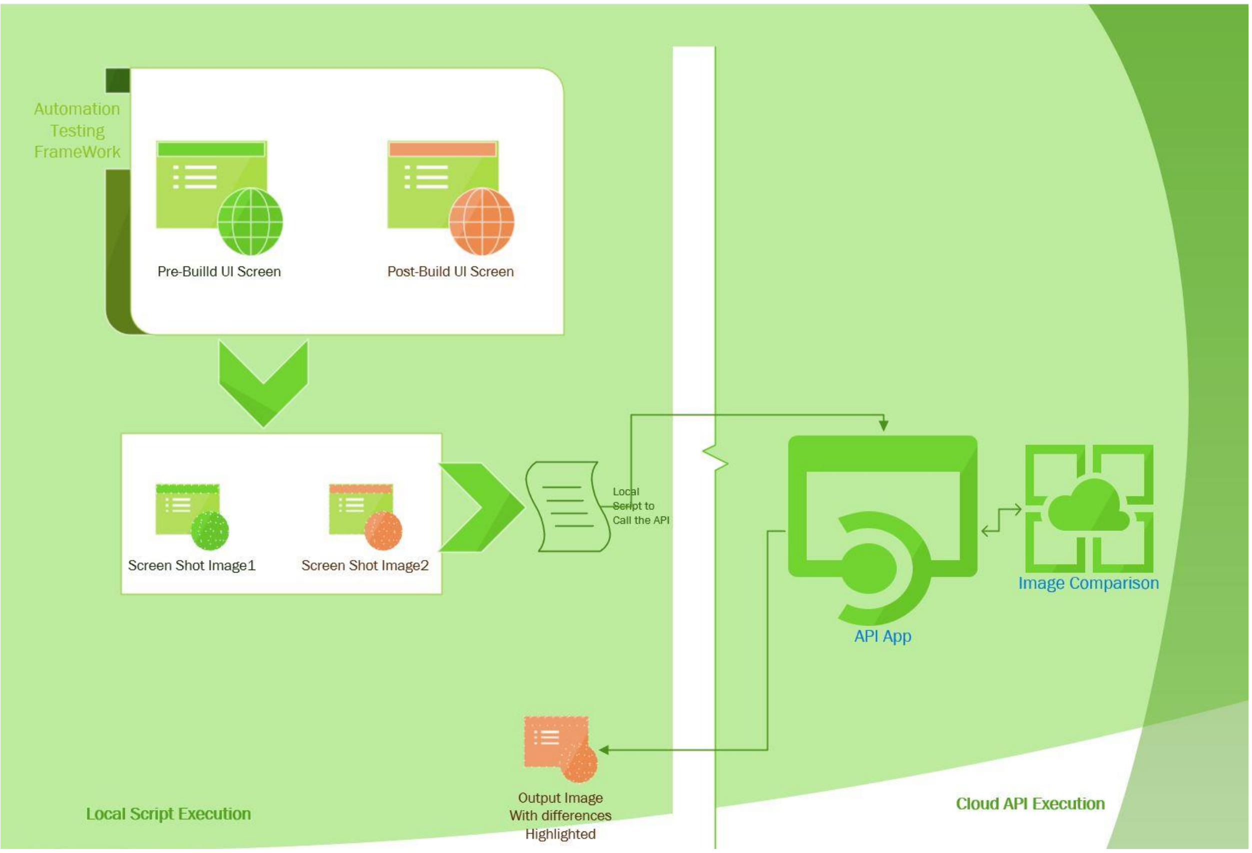
- Azure Notebooks was used for essential code development of the Python code
- Visual Studio was used to build the code locally and later deploy it

Azure Cloud Services

Used for hosting the solution

- Python Flask WebApp was created on Azure Cloud and hosted on an Ubuntu 16.04 Host
- Azure DevOps was used for Continuous Integration and Continuous Deployment of the Python-Flask application

Architecture Diagram


Automation Testing FrameWork
Pre-Buılld UI Screen
Post-Build UI Screen
Screen Shot Image1
Screen Shot Image2
Local Script to Call the API
API App
Image Comparison
Output Image With differences Highlighted
Local Script Execution
Cloud API Execution