USING TENSORFLOW AND COMPUTER VISION TO TEST GENERIC WEB SERVICE AVAILABILITY

Presented by Enrico La Vela
Agenda

• Brief intro to our test platform: ThEIA
• Use case: Computer Vision for testing web services
• Sikuli – brief explanation and problem faced
• Solution: Tensorflow – brief explanation
• Our Custom Object Detector applied to Login forms
• Results and final considerations
• Next steps
ThEIA Platform
Testing Environment for Internet Application

Services
OTT
- Video
- Social Networking
- VoIP
- Web application
- Unified Collaboration

Access Networks
- Fixed
- Wireless
- Mobile

ThEIA Manager

ThEIA Agent SW

Appliance agent PLUS

Appliance agent RACK

Appliance agent LITE
**Example of architecture:**

- Independent or coordinated tests managed with *Testplans*
ThEIA Platform  
Testing Environment for Internet Application

• One of ThEIA use cases:
  • **Computer Vision** for testing web services

• Agent acts as an user that access to web services and performs several actions
• Agent measures service availability, web pages responsiveness time, etc
How do we implement Computer Vision?

![Sikuli Image]

User Conference on Advanced Automated Testing

© All rights reserved
What’s Sikuli?

- **SikuliX** automates anything you see on the screen of your desktop computer.
- Sikuli uses image recognition powered by OpenCV to identify and control GUI components.
Why Sikuli?

• Sikuli is powerful in cases when there is no easy access to a GUI's internals or the source code of the application or web page you want to act on or when you want effectively test what the user sees on the screen -> Quality of Experience (QoE)

• It uses pixel detection in order to automate things

• We use Sikuli for testing QoE of web services and their availability through our ThEIA platform
How Sikuli works?

- It works under Java (> 8) environment and makes use of python scripts (2.7)
Limitations

• Sikuli is based on pixel detection

• When webpages’ GUI changes, Sikuli script fails
  -> test fails -> false negative results
  • Continuous changes in webpage layout, icons, button style

Example:

• You need to perform changes on script code and image capture
How to solve it?

Machine Learning

TensorFlow
What is TensorFlow?

• TensorFlow is an open source computational framework designed by the Google team, used to build **Machine Learning** models.
• It includes a feature of that defines, optimizes and calculates mathematical expressions easily with the help of multi-dimensional arrays called **tensors**.
• TensorFlow provides stable **Python** and **C++** APIs.
• There are a certain number of different models.
How do we used TensorFlow?

- We used the TensorFlow (Custom) Object Detection API (based on Python 3.6), invoked by our Sikuli scripts (based on Python 2.7)
- We have customized the model to recognize login forms, by Training and Evaluating a Custom Object Detector
Our custom *Login forms* Object Detector

**STEP 1**

- Create a dataset:
  - Go into several web pages with login forms and take screenshots (we used 700+ references for training phase)
- Random web page research (different domains)
- Language: English, Italian
Our custom *Login forms* Object Detector

**STEP 2**

- Label each image using a graphic image annotation tool
  
  ![LabelImg](image)

- **LabelImg** generates XML files with annotation

- Convert XML files into a unified CSV file
Our custom *Login forms* Object Detector

**STEP 3**

- Generate **TFRecords**

- TensorFlow object detection API doesn’t take csv files as an input, but it needs *tf record files* to train the model

- Use of a python scripts to generate *record file* from *CSV file*
Our custom *Login forms* Object Detector

**STEP 4 - [1/2]**

- Training the Model
- Use of a pre-trained model as starting point:
  - `faster_rcnn_inception_v2_coco`
  - i.e. Faster R-CNN with Inception v2 algorithm for MSCOCO Dataset
- **What are these acronyms?**
  - They are related to algorithm and dataset used
  - R-CNN: **Region-Convolutional Neural Network**

---

What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints
Our custom *Login forms* Object Detector

STEP 4 - [2/2]

- Training the Model
- With a normal laptop: spent 1 week in computation
- ~41K training steps
Our custom *Login forms* Object Detector

**STEP 5**

- Evaluate the Model
- Use of a different data set (not the 700+ screenshots, but others...~70)
- Some iteration in training phase (STEP 4)
Our custom *Login forms* Object Detector

**STEP 6**

- The model is well trained and results are acceptable
  - Training phase is complete

- It is possible to use the model database (frozen inference graph `.pb` file) to perform the *login forms* object detection with your py scripts
  - In our PoC case the `.pb` file is ~60MB
Results [1/2]

• Our scripts are able to perform login access to almost any web page (below a real example)
Results [2/2]

- We have optimized our Sikuli scripts, making use of some logic (application layer) that helps to detect typical position of text fields, two steps form and Login button (use of OCR)
Final considerations

• Lightweight py scripts executed at client side that avoid script rework when web pages changes

• Limitations: it doesn’t work with captcha

• It is a Proof of Concept (PoC), still limited to perform generic login action
Next steps

• Apply the same concept (Sikuli + TensorFlow API) to other kind of web page common actions (post a comment, upload a photo, ...)

• Apply TensorFlow API to evaluate QoE of video content (i.e. MOS evaluation)
Q&A

• Authors Contacts
  • Enrico La Vela e.lavela@netresults.it [SPEAKER]
  • Silvia Vistoli s.vistoli@netresults.it
  • Sergio Borghese s.borghese@netresults.it
  • Francesco Oppedisano f.oppedisano@netresults.it