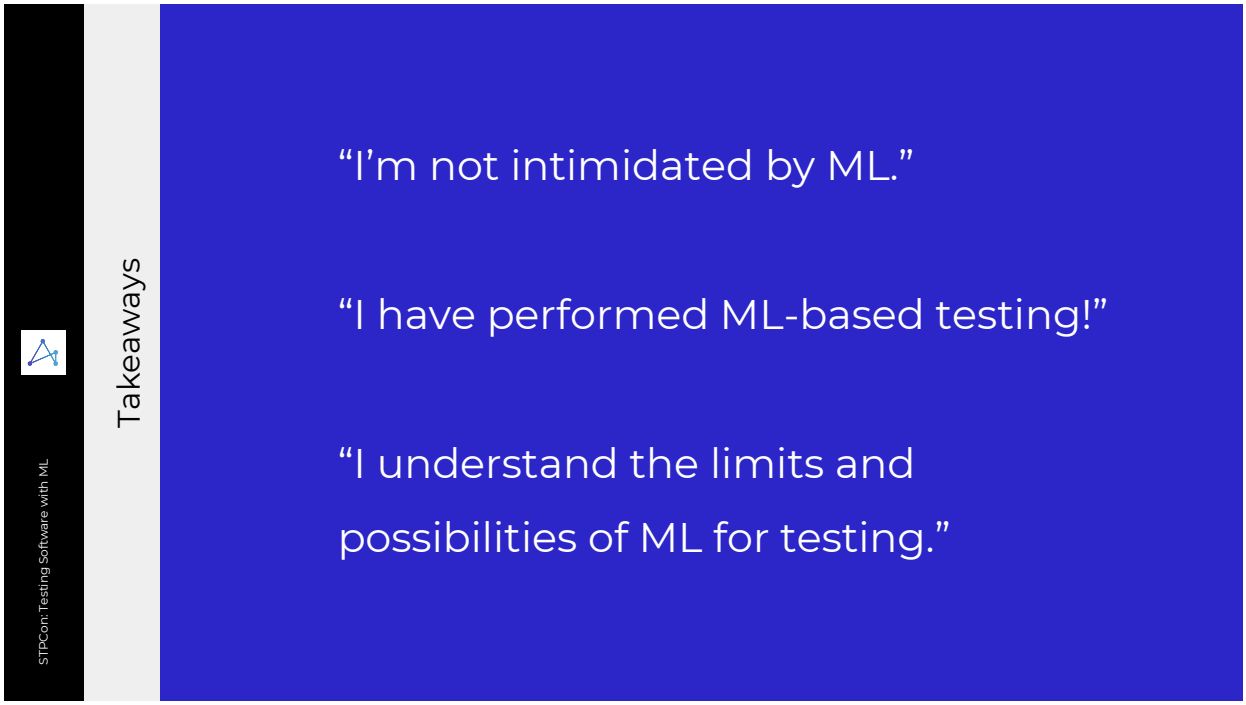

SOFTWARE
TEST PROFESSIONALS
CONFERENCE SPRING 2019
Testing Software with ML
Jason Arbon

1


STPCon: Testing Software with ML

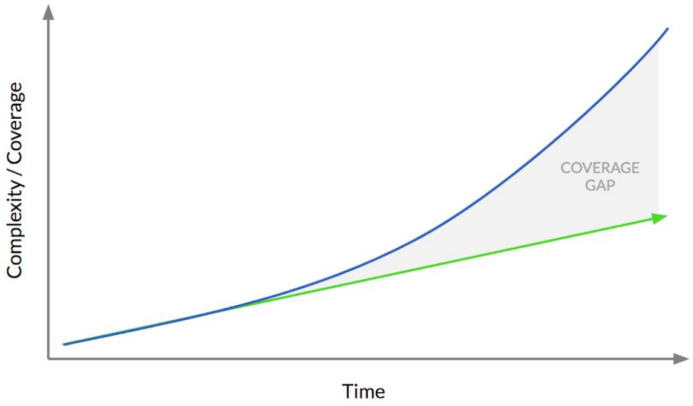
2


Takeaways
STPCon: Testing Software with ML
"I'm not intimidated by ML."
"I have performed ML-based testing!"
"I understand the limits and possibilities of ML for testing."

3


STPCon: Testing Software with ML
Why ML?

4



Features
Complexity increases exponentially as new features and states combine

Tests
Test coverage grows linearly because tests can only be added one at a time

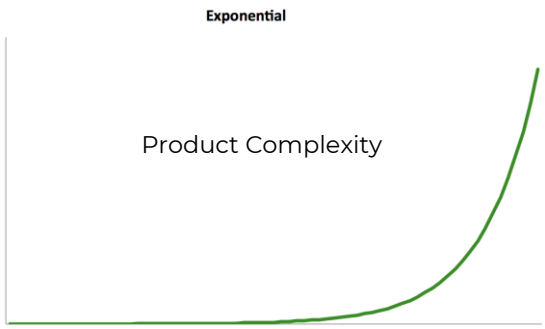
5

Desktop

Web

Mobile

Cloud



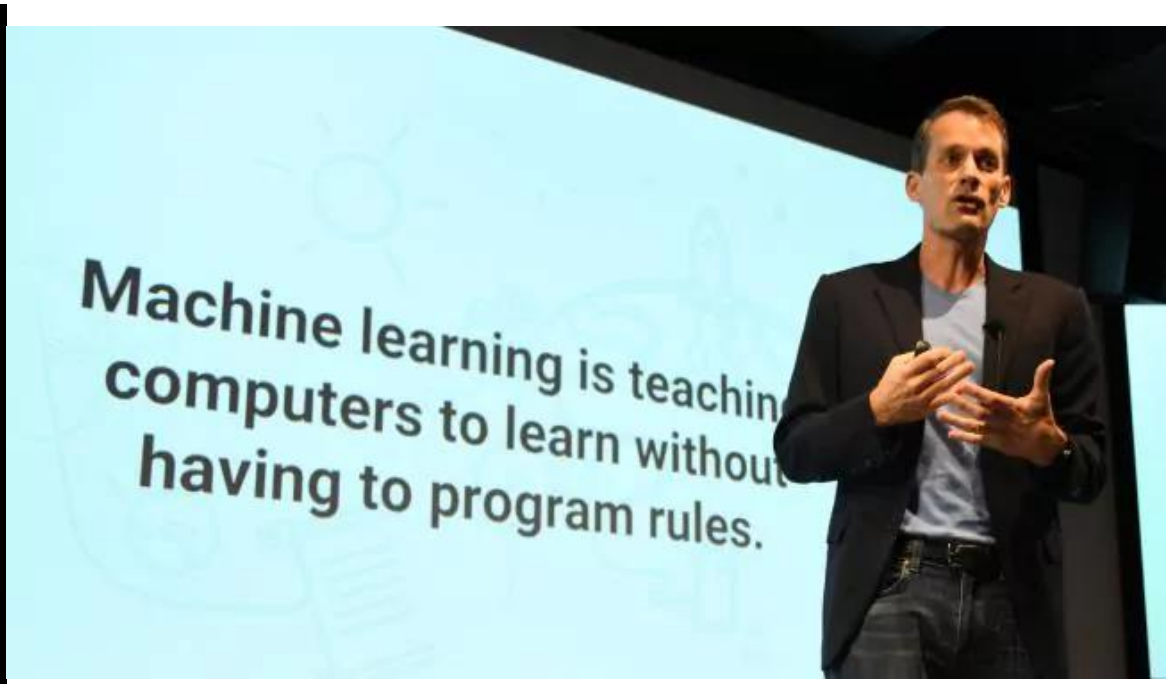
6

ML

Replicates Human Actions

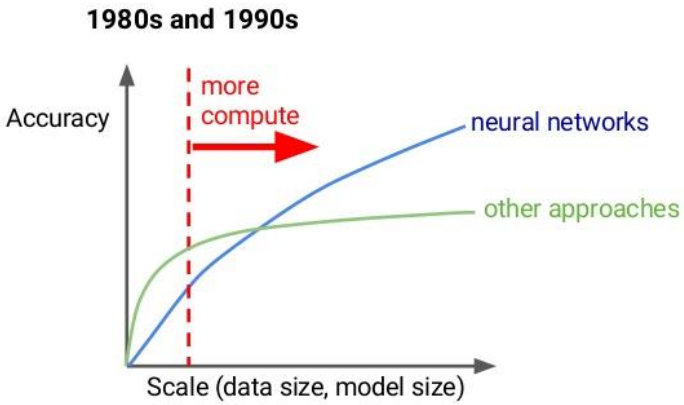
Finally a tech to HELP Testing

7



8

Why Now?



--Jeff Dean, AI Frontiers

9

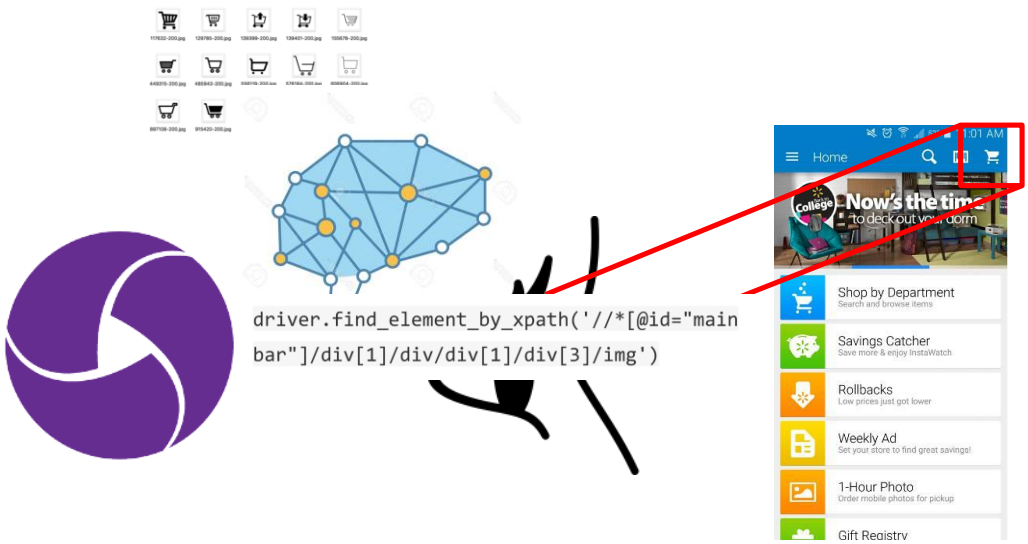
Element Selection

10

Human

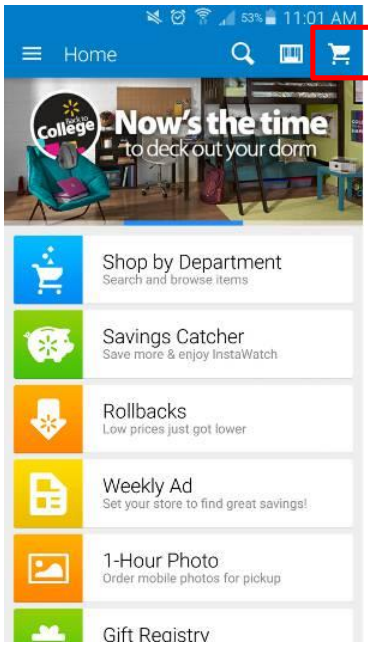
- Looks at Screen
- Sees all the ‘Elements’
- Identifies elements by name/function
- Decides what to do next…

11



12

Object Detection

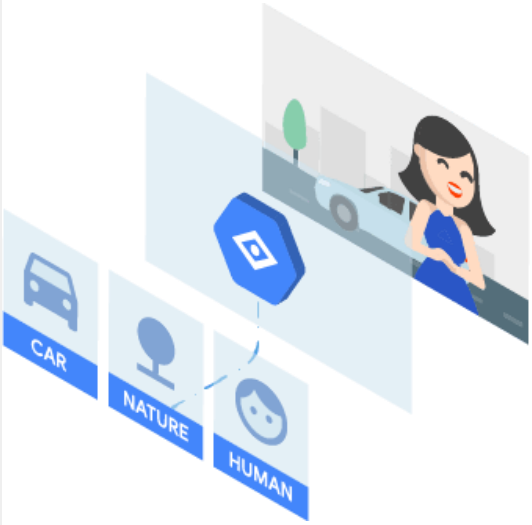


?

13

Objects Detection

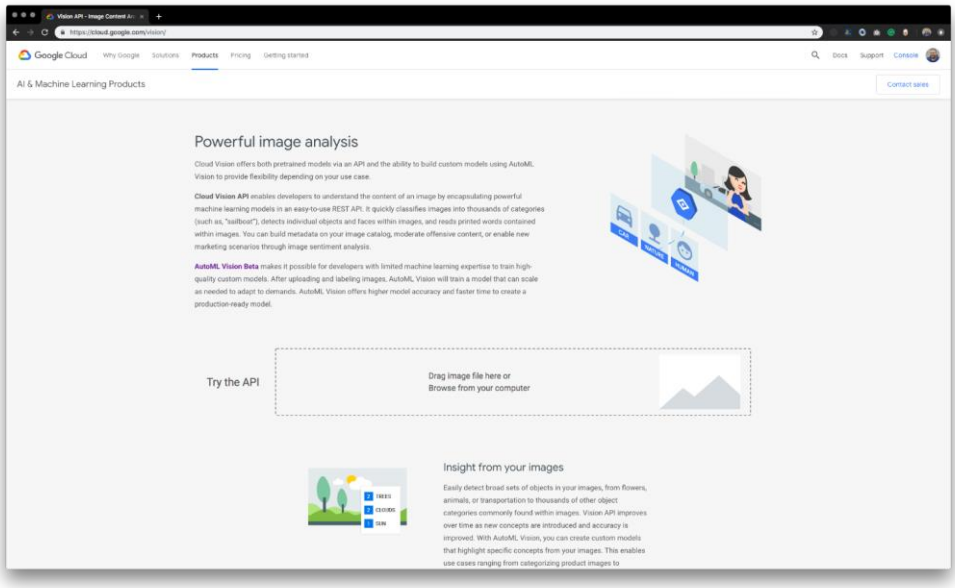
CLOUD AUTOML VISION



14

AutoML

https://cloud.google.com/vision/



15

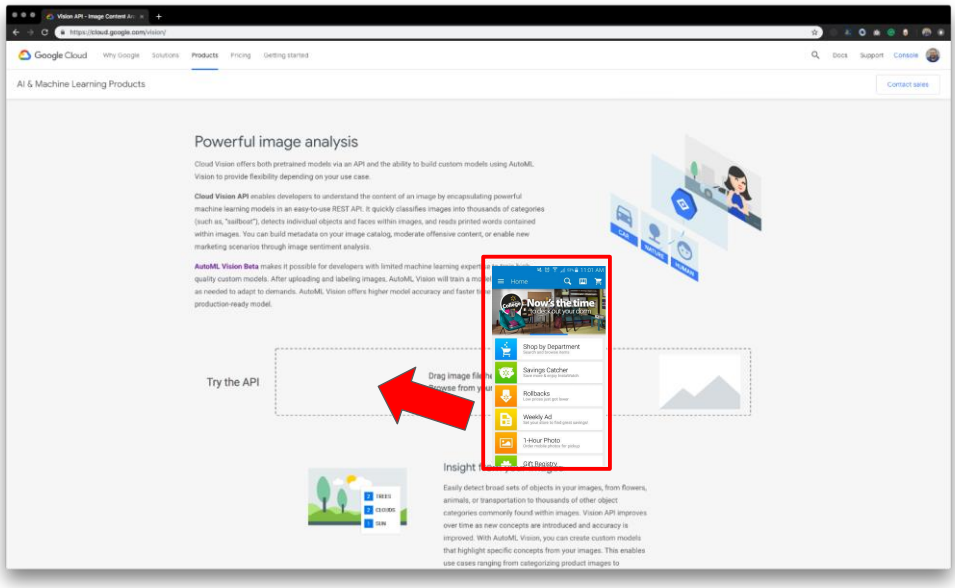
STPCon: Testing Software with ML

- Object Detection*
- Label Detection
- Web Locations
- Optical Character Recognition
- Handwriting Recognition
- Logo Detection
- Landmarks, Faces, Content Moderation
- REST API

16

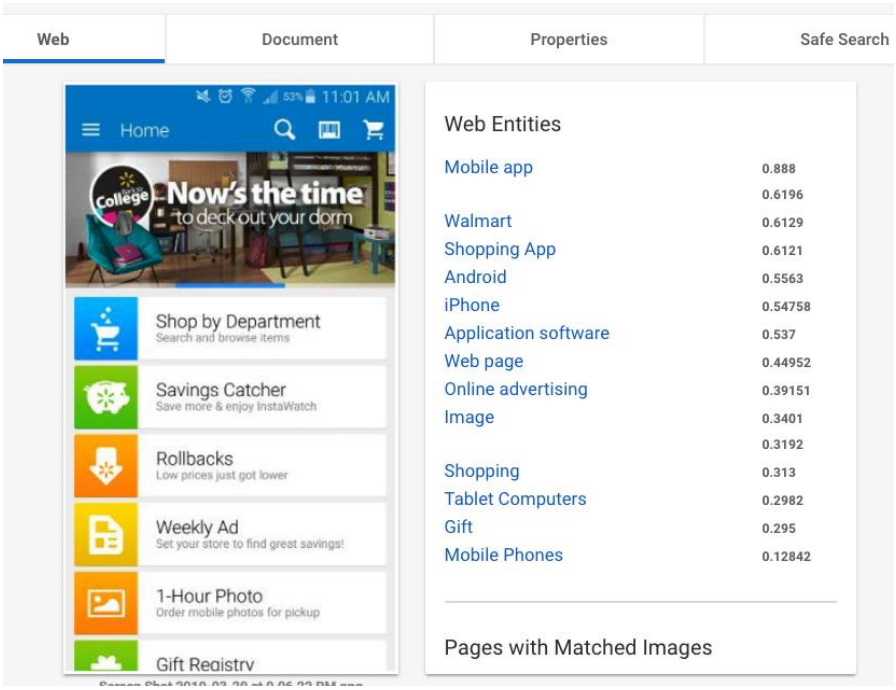
AutoML

Upload Screenshot



17

Web



18

Questions

Are you surprised the ML figured out this was the Walmart app?

19

Questions

Are you surprised the ML realized this was a mobile app/screenshot?

20

Questions

Why do you think the ML Thinks this is ‘Online Advertising’?

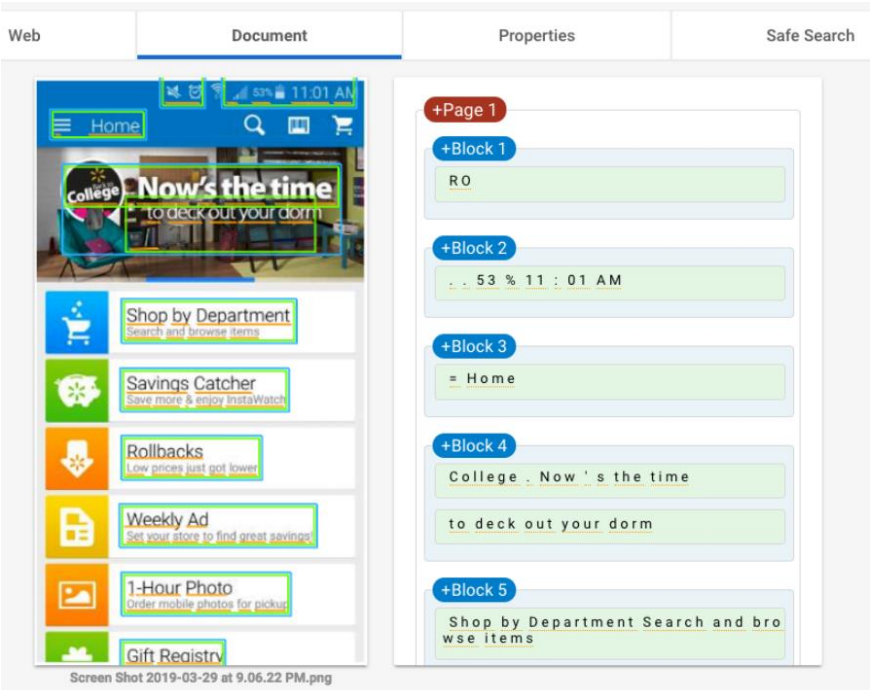
21

Questions

What did the ML miss?

22

Document



23

Questions

Are you impressed that the ML could find so many elements from just a Screenshot? similar to the human eye/brain?

24

Questions

What types of elements did the ML recognize that your brain did not?

25

Questions

What types of elements did the ML recognize that a typical code/DOM-based analysis might have missed?

26

Questions

What types of elements did the ML miss, and why do you think it missed them?

27

Questions

How could this data help “Manual Testing”?

28

Questions

Did the ML find the shopping cart as we'd hoped?

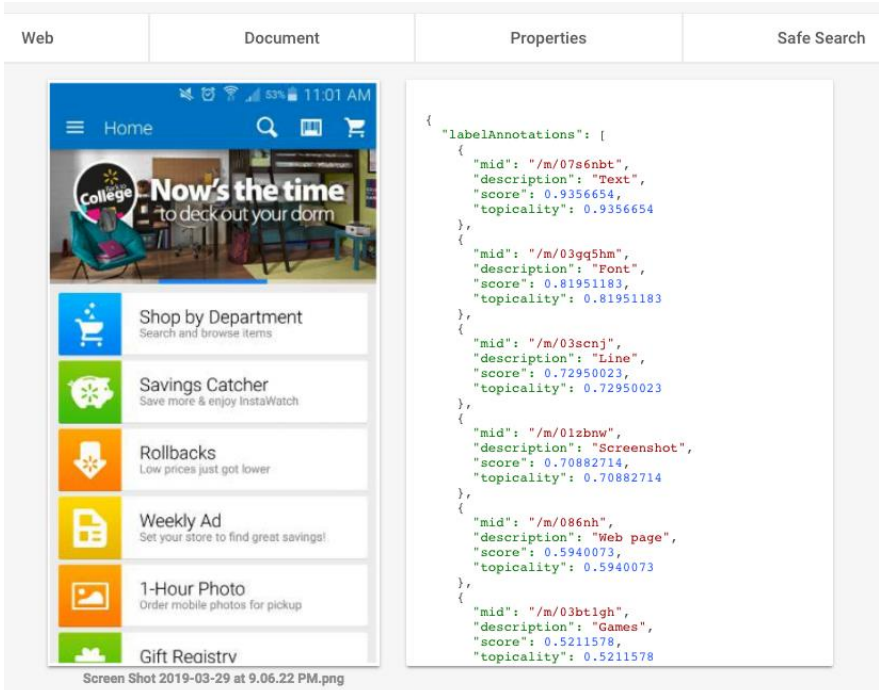
29

Questions

How could this ML response be useful in testing even if it didn't find the shopping cart, or 'every element'?

30

JSON



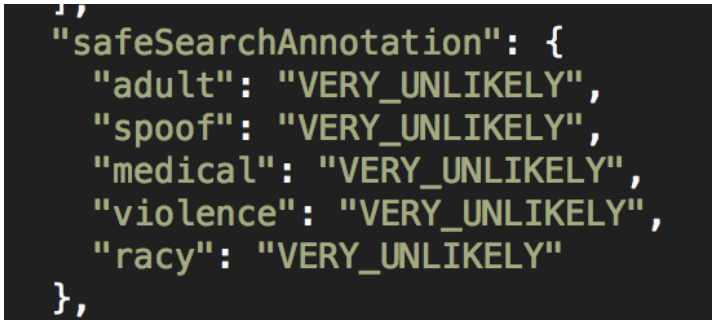
31

JSON

```
{
  "description": "Gift",
  "boundingPoly": {
    "vertices": [
      {
        "x": 95,
        "y": 590
      },
      {
        "x": 123,
        "y": 590
      },
      {
        "x": 123,
        "y": 609
      },
      {
        "x": 95,
        "y": 609
      }
    ]
  }
},
```

32

Safe Search



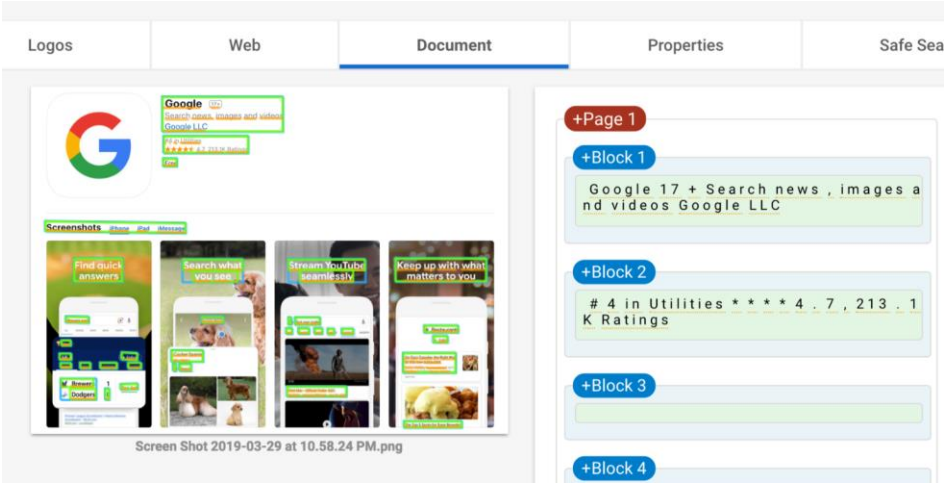
33

Questions

How could safe search help app testing?

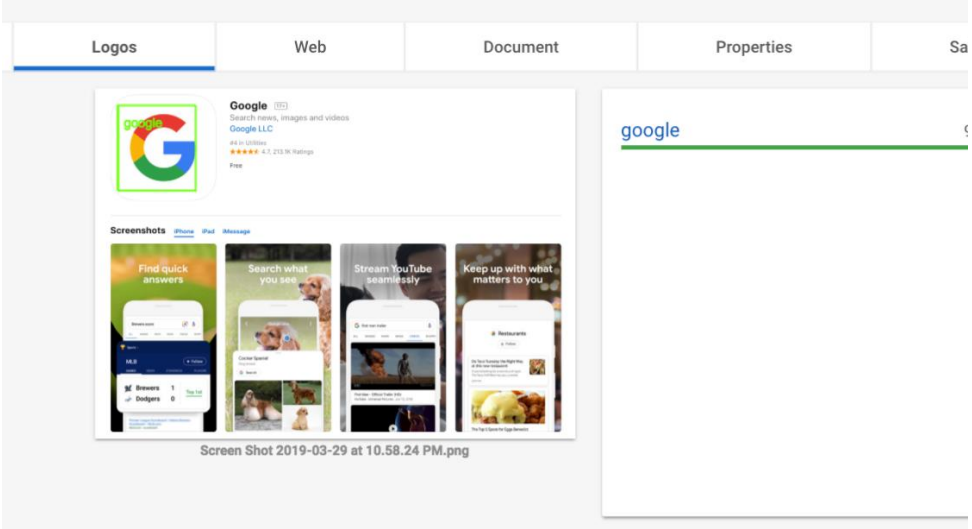
34

Google App



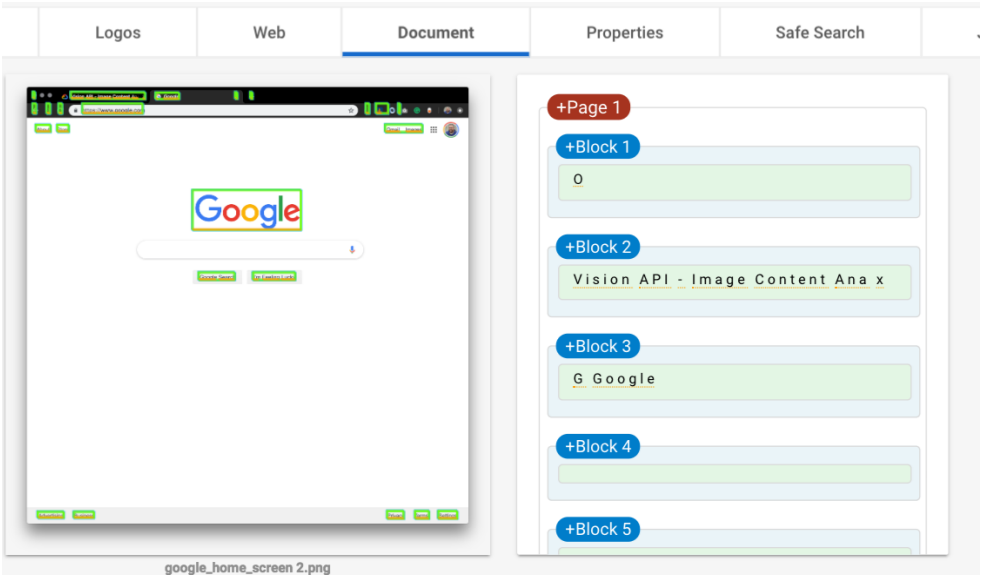
35

Logos



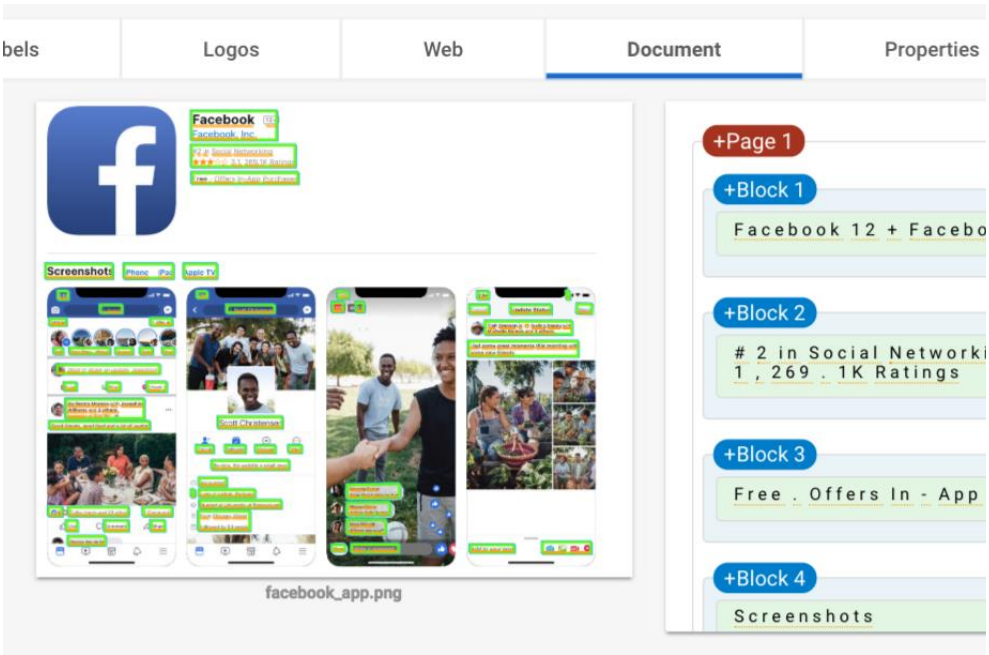
36

Web



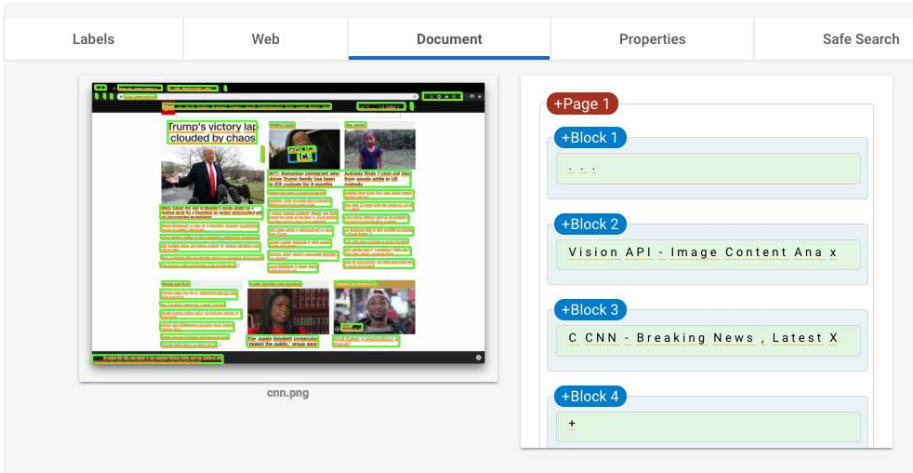
37

Facebook App



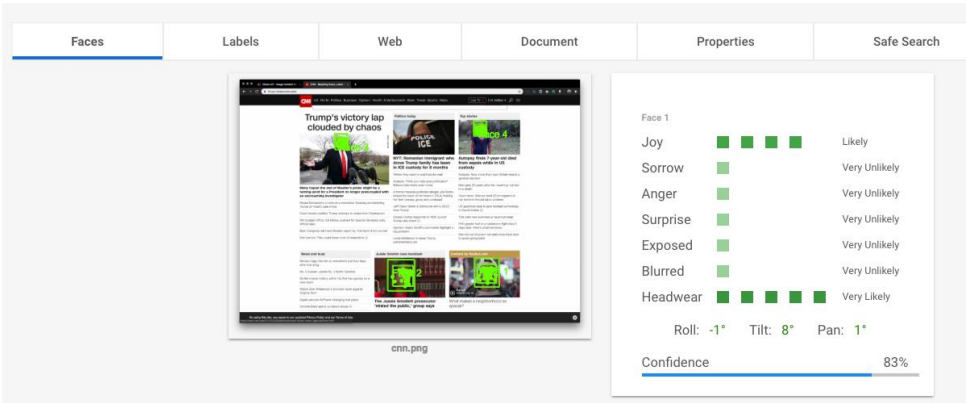
38

CNN



39

Face Detection



40

15 Min

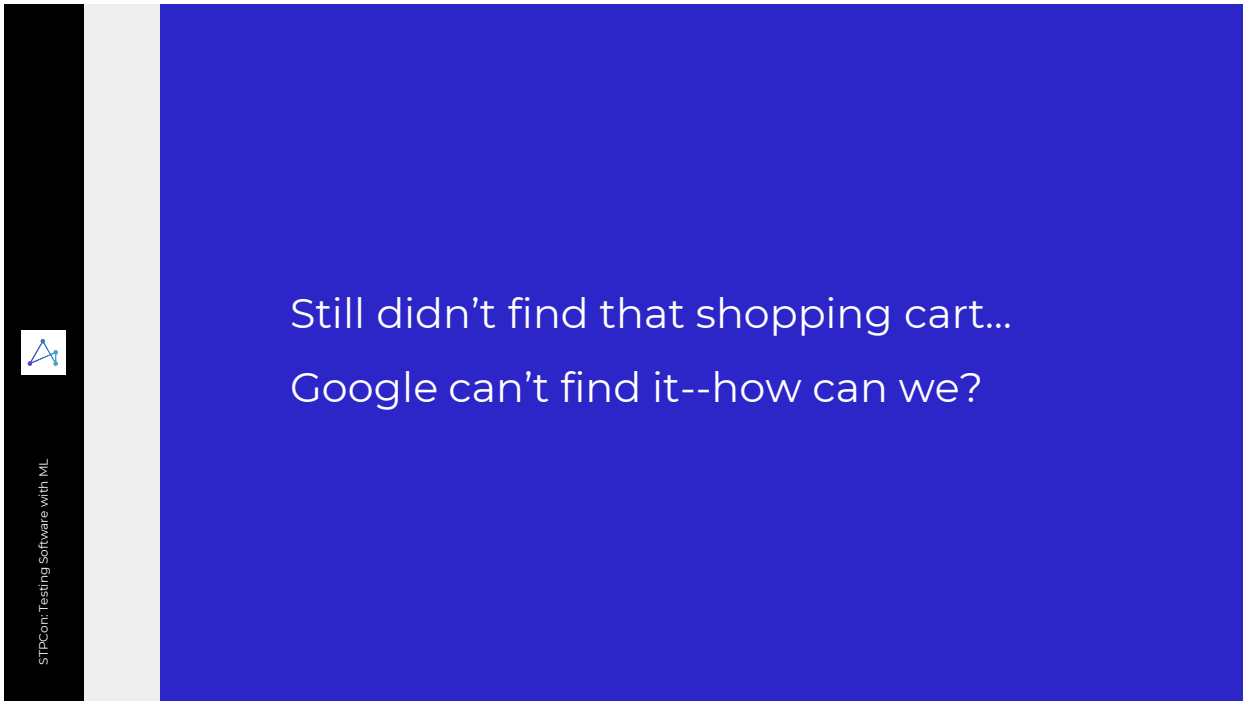
<Your App Here>

41

Exercise

Share thoughts Applying ML to your app!

42

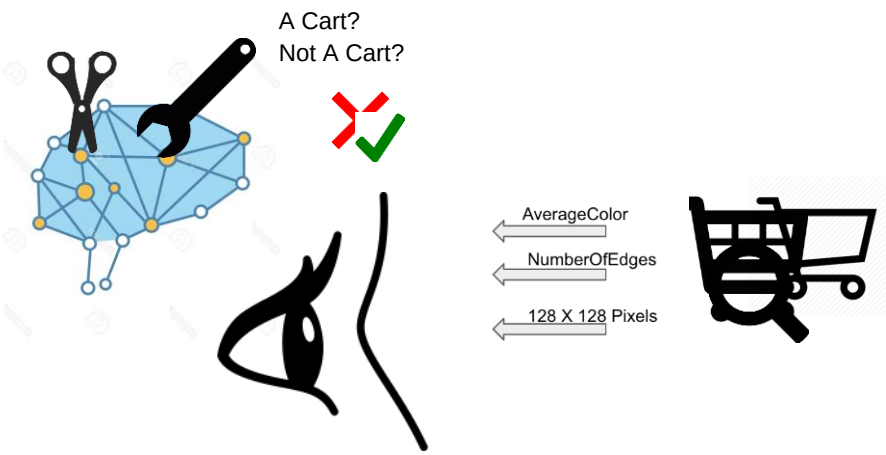

STPCon: Testing Software with ML
Still didn't find that shopping cart...
Google can't find it--how can we?

43

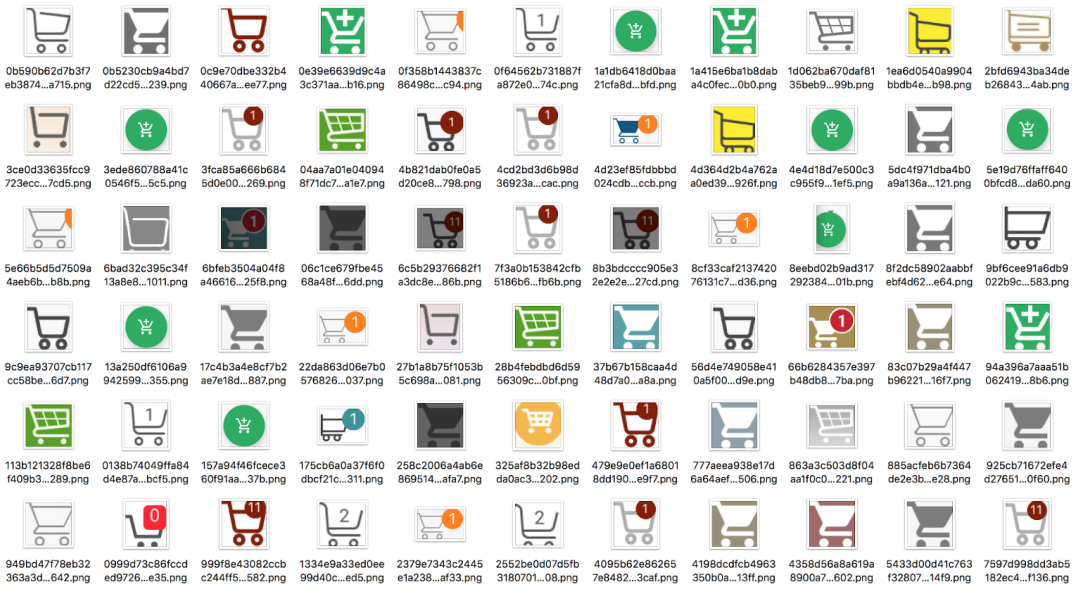

STPCon: Testing Software with ML
We build our 'own' ML for testing

44

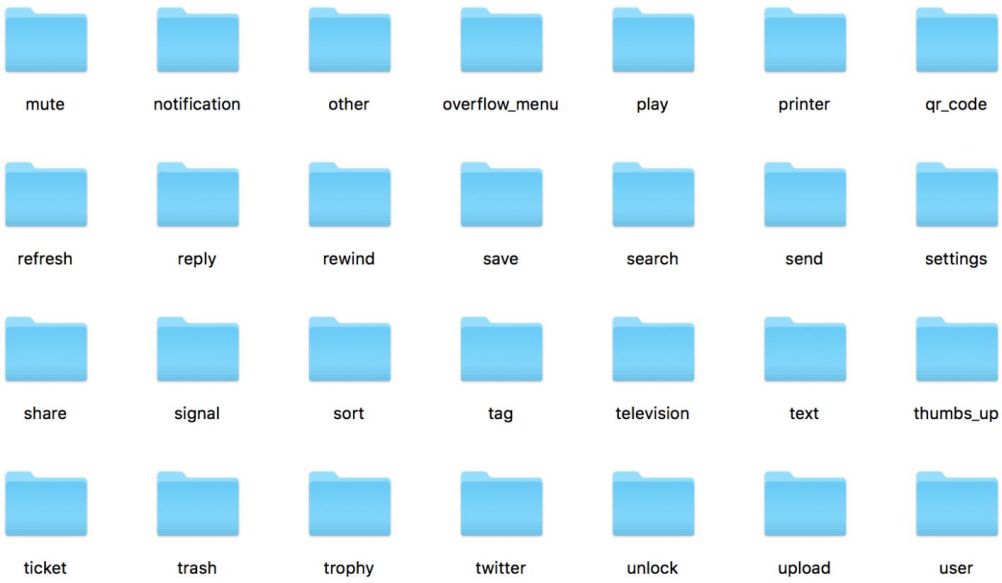
Train Guess/Test Features Images



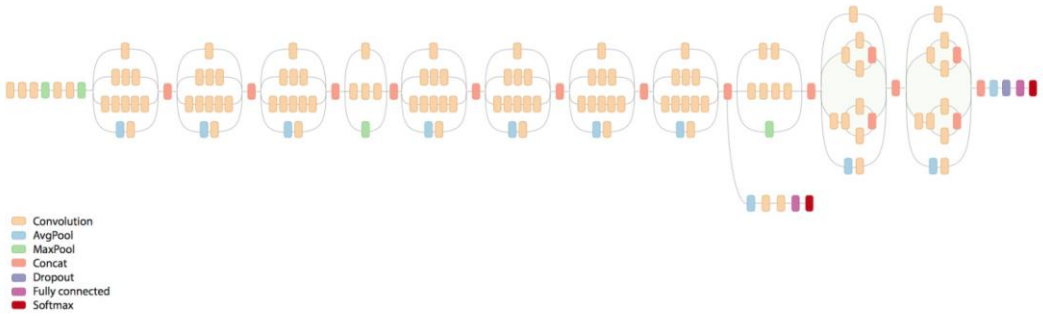
45



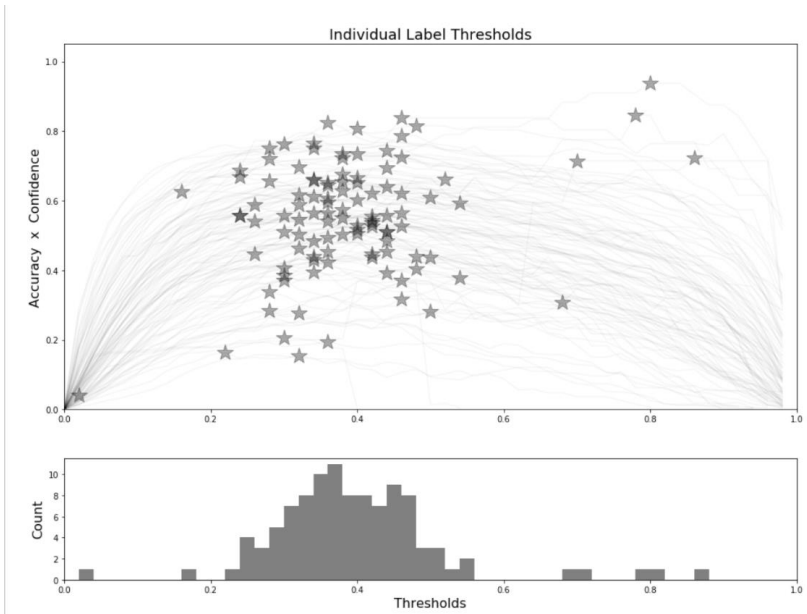
46


mute
notification
other
overflow_menu
play
printer
qr_code
refresh
reply
rewind
save
search
send
settings
share
signal
sort
tag
television
text
thumbs_up
ticket
trash
trophy
twitter
unlock
upload
user


TensorFlow

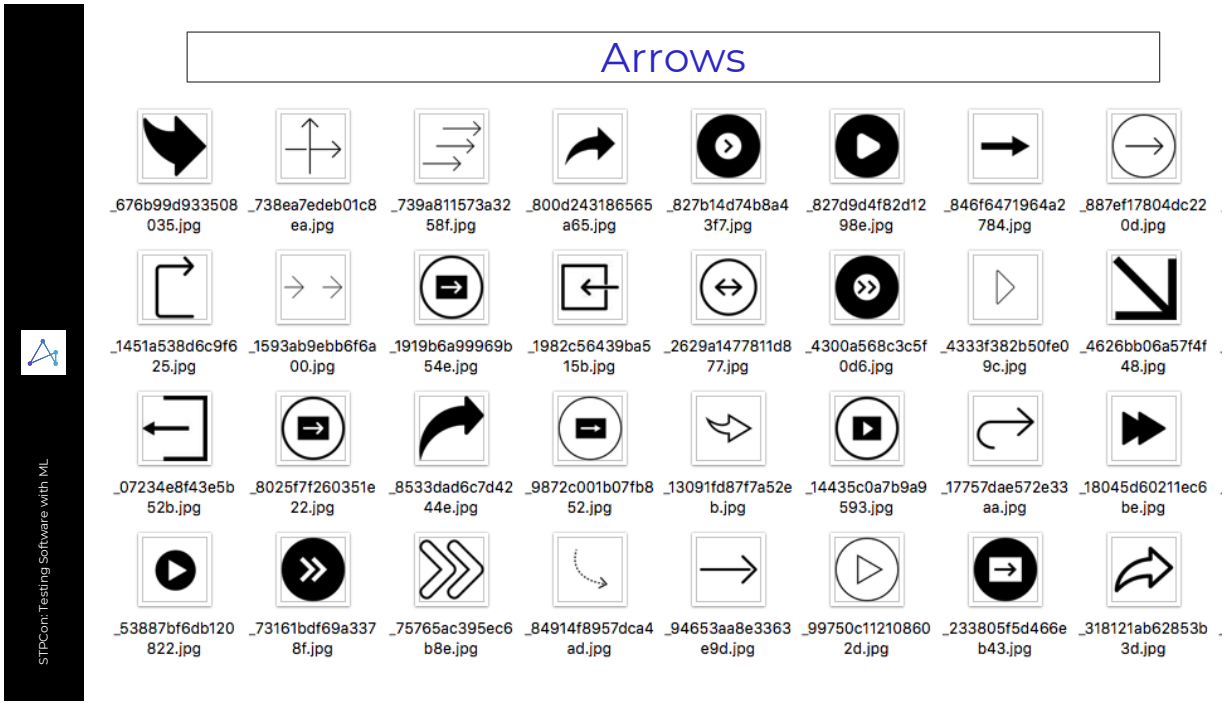


49

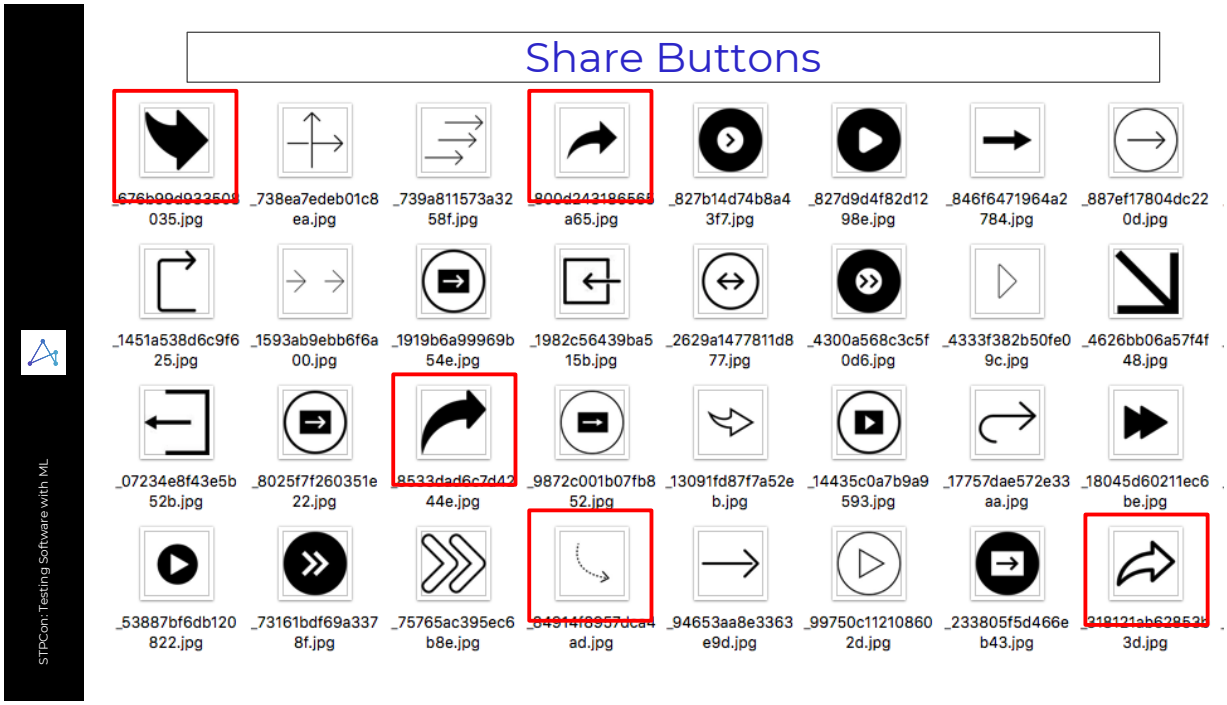


	thresh	conf	accu
fingerprint:	0.8000	1.0000	0.9375
google:	0.7800	1.0000	0.8462
fab:	0.4600	0.9000	0.9310
barcode:	0.3600	0.8767	0.9412
eye:	0.4800	0.9459	0.8621
thumbs_up:	0.4000	0.9478	0.8516
trophy:	0.4600	0.9466	0.8322
headphones:	0.3400	0.9213	0.8305
car:	0.3000	0.9373	0.8154
home:	0.2800	0.9405	0.8000
calculator:	0.3400	0.8875	0.8458
globe:	0.4400	0.9362	0.7942
cart:	0.4000	0.9539	0.7708
airplane:	0.3800	0.9388	0.7814
facebook:	0.4600	0.8222	0.8810
light_bulb:	0.3800	0.9595	0.7532
profile_avatar:	0.8600	0.8500	0.8500
qr_code:	0.2800	0.8505	0.8476
twitter:	0.7000	1.0000	0.7143
camera:	0.3200	0.9419	0.7390
...			
rewind:	0.3600	0.7692	0.5511
minimize:	0.3000	0.8000	0.5094
sort:	0.4800	0.7108	0.5673
fast_forward:	0.3400	0.6092	0.6486
download:	0.4400	0.6154	0.6371
mute:	0.3000	0.7068	0.5446
overflow_menu:	0.5400	0.6154	0.6154
arrow_left:	0.3000	0.7746	0.4802
attach:	0.4600	0.4332	0.8571
reply:	0.2800	0.6300	0.5354
maximize:	0.4600	0.6984	0.4536
bag:	0.6800	0.7714	0.3971
chart:	0.2800	0.6286	0.4531
external_link:	0.5000	0.6667	0.4211
alarm:	0.3200	0.6495	0.4254
inbox:	0.3000	0.3902	0.5291
delete:	0.3600	0.5350	0.3627
arrow_right:	0.2200	0.3714	0.4375
upload:	0.3200	0.2698	0.5690
gmail:	0.0200	0.3333	0.1250

50


Arrows
_676b99d933508035.jpg
_738ea7edeb01c8ea.jpg
_739a811573a3258f.jpg
_800d243186565a65.jpg
_827b14d74b8a43f7.jpg
_827d9d4f82d1298e.jpg
_846f6471964a2784.jpg
_887ef17804dc220d.jpg
_1451a538d6c9f625.jpg
_1593ab9ebb6f6a00.jpg
_1919b6a99969b54e.jpg
_1982c56439ba515b.jpg
_2629a1477811d877.jpg
_4300a568c3c5f0d6.jpg
_4333f382b50fe09c.jpg
_4626bb06a57f4f48.jpg
_07234e8f43e5b52b.jpg
_8025f7f260351e22.jpg
_8533dad6c7d4244e.jpg
_9872c001b07fb852.jpg
_13091fd87f7a52eb.jpg
_14435c0a7b9a9593.jpg
_17757dae572e33aa.jpg
_18045d60211ec6be.jpg
_53887bf6db120822.jpg
_73161bdf69a3378f.jpg
_75765ac395ec6b8e.jpg
_84914f8957dca4ad.jpg
_94653aa8e3363e9d.jpg
_99750c11210860 2d.jpg
_233805f5d466eb43.jpg
_318121ab62853b3d.jpg
STPCon: Testing Software with ML

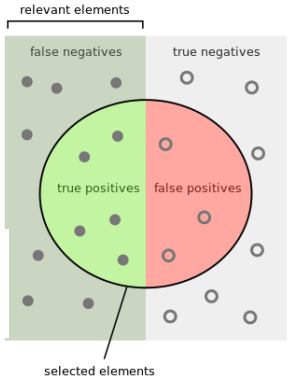
51

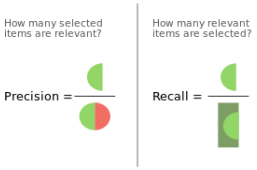

Share Buttons
_676b99d933508035.jpg
_738ea7edeb01c8ea.jpg
_739a811573a3258f.jpg
_800d243186565a65.jpg
_827b14d74b8a43f7.jpg
_827d9d4f82d1298e.jpg
_846f6471964a2784.jpg
_887ef17804dc220d.jpg
_1451a538d6c9f625.jpg
_1593ab9ebb6f6a00.jpg
_1919b6a99969b54e.jpg
_1982c56439ba515b.jpg
_2629a1477811d877.jpg
_4300a568c3c5f0d6.jpg
_4333f382b50fe09c.jpg
_4626bb06a57f4f48.jpg
_07234e8f43e5b52b.jpg
_8025f7f260351e22.jpg
_8533dad6c7d4244e.jpg
_9872c001b07fb852.jpg
_13091fd87f7a52eb.jpg
_14435c0a7b9a9593.jpg
_17757dae572e33aa.jpg
_18045d60211ec6be.jpg
_53887bf6db120822.jpg
_73161bdf69a3378f.jpg
_75765ac395ec6b8e.jpg
_84914f8957dca4ad.jpg
_94653aa8e3363e9d.jpg
_99750c11210860 2d.jpg
_233805f5d466eb43.jpg
_318121ab62853b3d.jpg
STPCon: Testing Software with ML

52

F1 Score (thanks wikipedia)

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$





53

F1 at test.ai Today

Test the classifier on 1747 vapor validation samples: F1 score (**97%**).

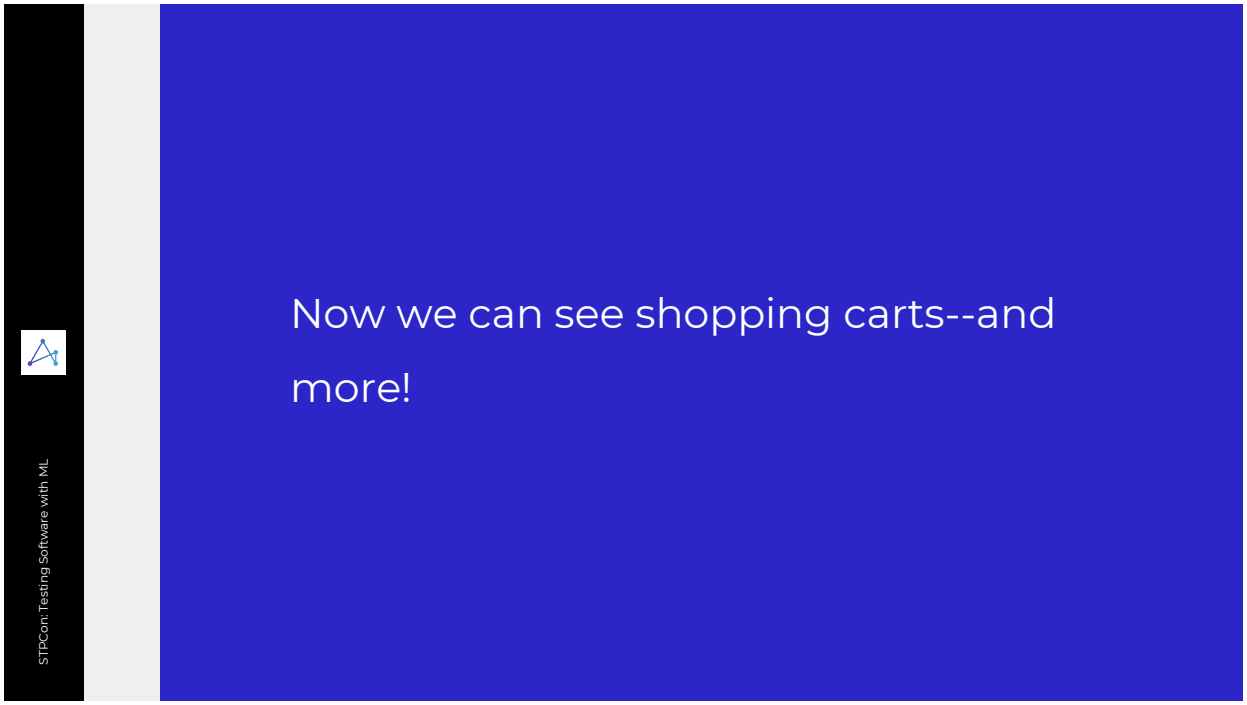
	precision	recall	f1-score	support
_negative	0.85	0.85	0.85	48
app_message_popup	0.99	0.95	0.97	78
article	0.96	0.91	0.94	176
home	0.99	0.98	0.99	524
login	0.98	0.99	0.99	187
menu_drawer	1.00	0.99	1.00	106
search	0.99	0.95	0.97	149
search_results	0.96	0.98	0.97	134
settings	1.00	1.00	1.00	76
sign_in_up_options	1.00	1.00	1.00	64
system_permissions_popup	1.00	1.00	1.00	205
micro avg	0.98	0.97	0.98	1747
macro avg	0.97	0.96	0.97	1747
weighted avg	0.98	0.97	0.98	1747
samples avg	0.97	0.97	0.97	1747

Test the classifier on 76 log-in screenshots that Yash collected: Accuracy: **97%**;

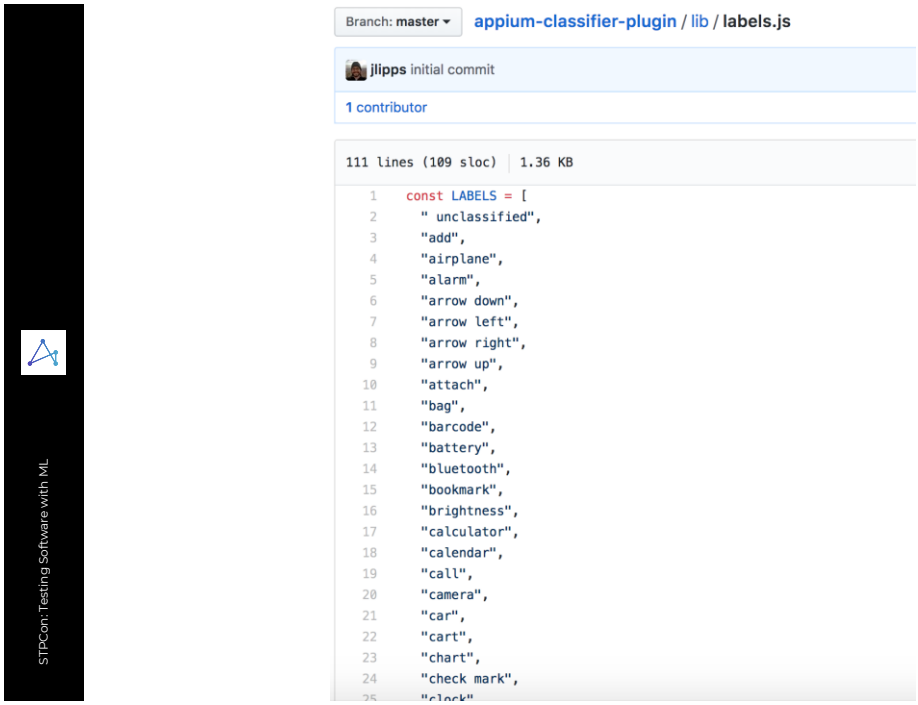
*Only **2** misclassified as 'sign_in_up_options' and 'unknown'.*

```
['login', 'login', 'login', 'login',
'sign_in_up_options', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'unknown',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login', 'login', 'login', 'login', 'login', 'login',
'login', 'login']
```

54


STPCon: Testing Software with ML
Now we can see shopping carts--and more!

55


STPCon: Testing Software with ML
Branch: master
appium-classifier-plugin / lib / labels.js
jlipps initial commit
1 contributor
111 lines (109 sloc) 1.36 KB
const LABELS = [
" unclassified",
"add",
"airplane",
"alarm",
"arrow down",
"arrow left",
"arrow right",
"arrow up",
"attach",
"bag",
"barcode",
"battery",
"bluetooth",
"bookmark",
"brightness",
"calculator",
"calendar",
"call",
"camera",
"car",
"cart",
"chart",
"check mark",

56

Open Source: Code and Data

Build Classifier

To build the classifier from the provided images run the following.

```
python retrain.py --image_dir training_images/ --output_graph output/saved_model.pb --output_labels
output/saved_model.pbtxt --how_many_training_steps 4000 --learning_rate 0.30 --testing_percentage 25 --
validation_percentage 25 --eval_step_interval 50 --train_batch_size 2000 --test_batch_size -1 --
validation_batch_size -1 --bottleneck_dir /tmp/bottleneck
```

Run Classifier

To use the classifier to classify images, run the script under `sample_run/` directory.

```
python run_model.py cart.png
```

57

Your App's Elements?

What does your app have that the open source classifier doesn't?

58

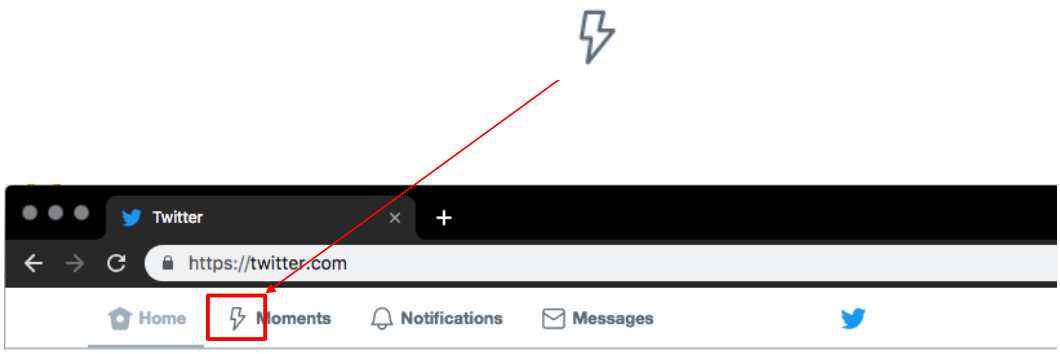
Your App's Elements?

1. Find An Element in your app that isn't in the standard list.
2. Get an image of this from your app.
3. Get similar images on the web

59

STPCon: Testing Software with ML

Add: "Moments/Lighting" Button

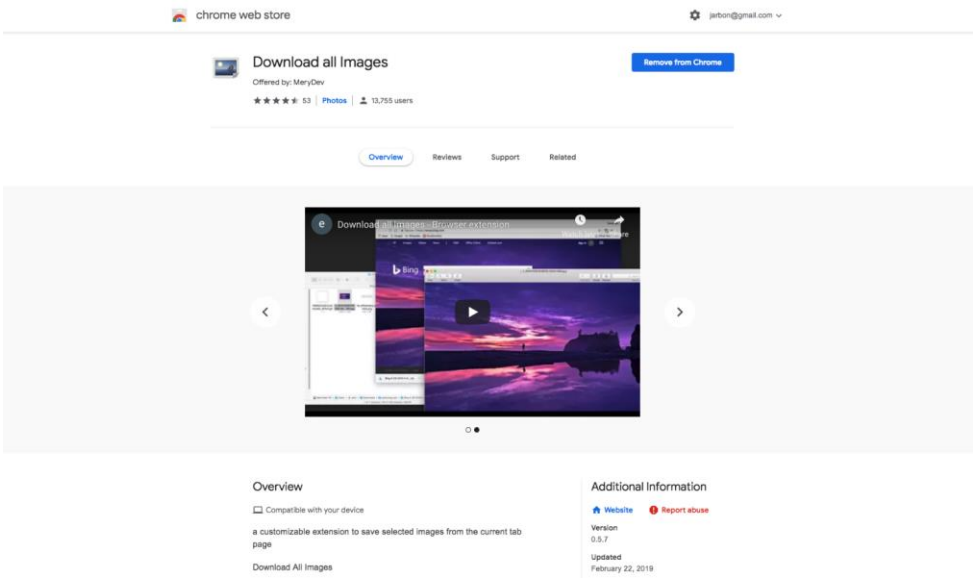


60

Search for Similar Icons on Web

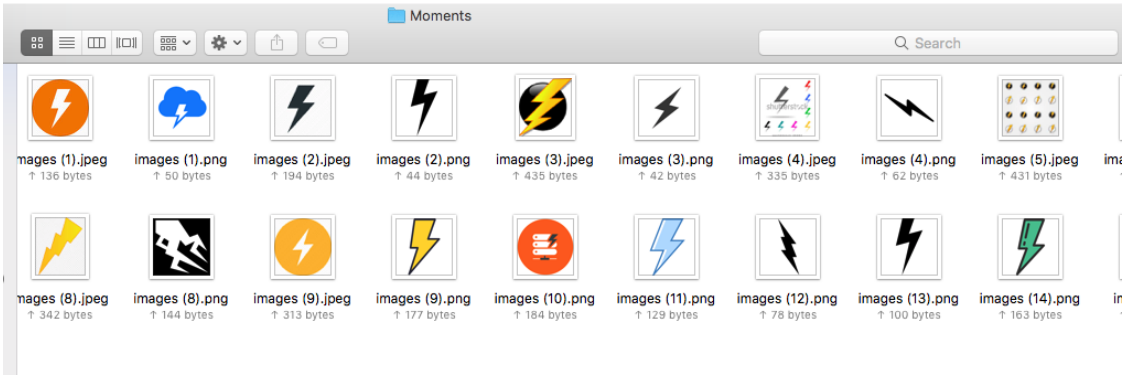
61

Tip: Extension to Download All

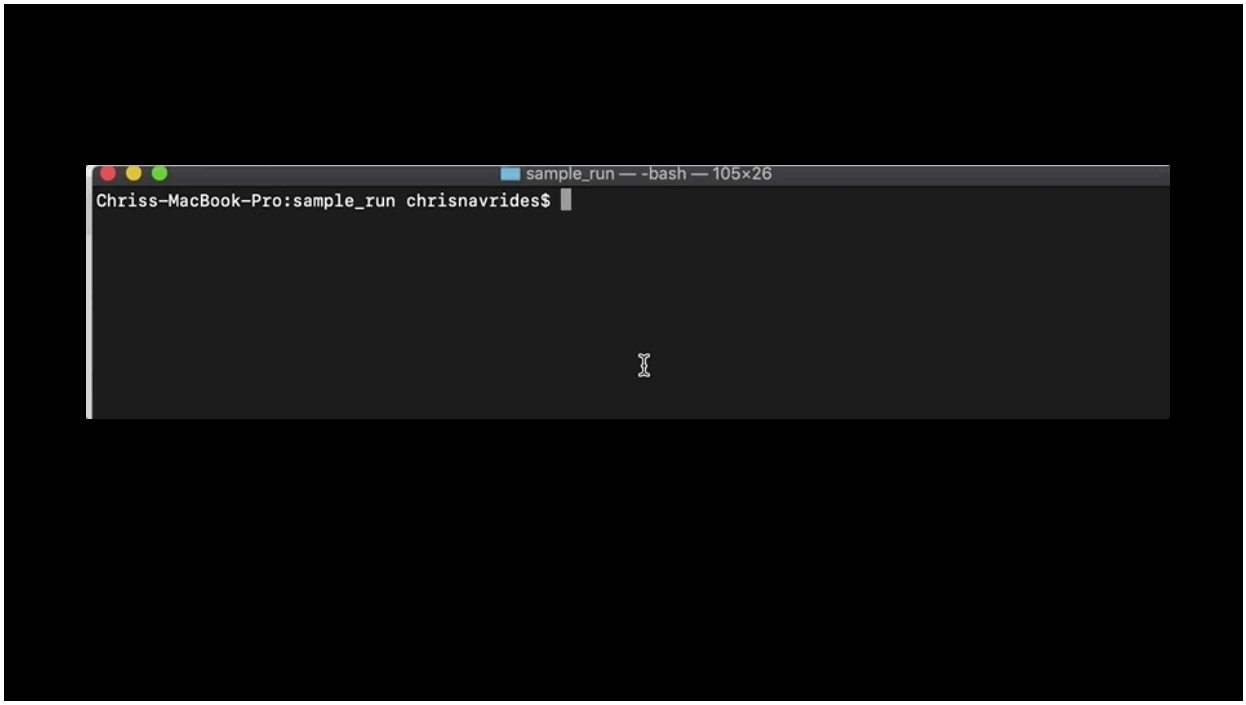


62

Add folder called “Moments” with Images



Later :)



65

Retrain the ML

Build Classifier

To build the classifier from the provided images run the following.

```
python retrain.py --image_dir training_images/ --output_graph output/saved_model.pb --output_labels output/saved_model.pbtxt --how_many_training_steps 4000 --learning_rate 0.30 --testing_percentage 25 --validation_percentage 25 --eval_step_interval 50 --train_batch_size 2000 --test_batch_size -1 --validation_batch_size -1 --bottleneck_dir /tmp/bottleneck
```

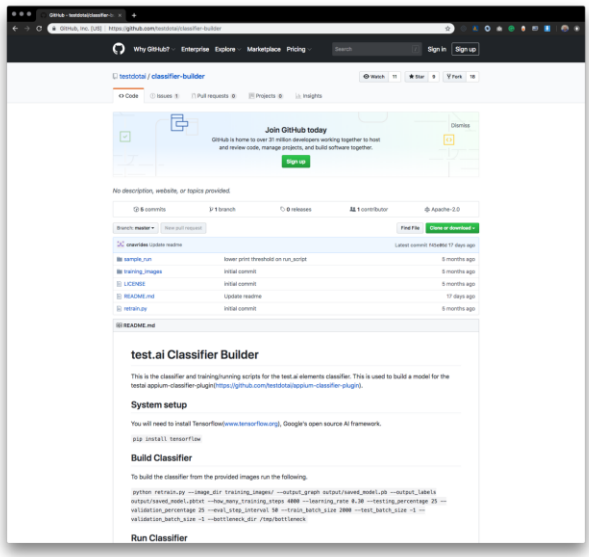
Run Classifier

To use the classifier to classify images, run the script under `sample_run/` directory.

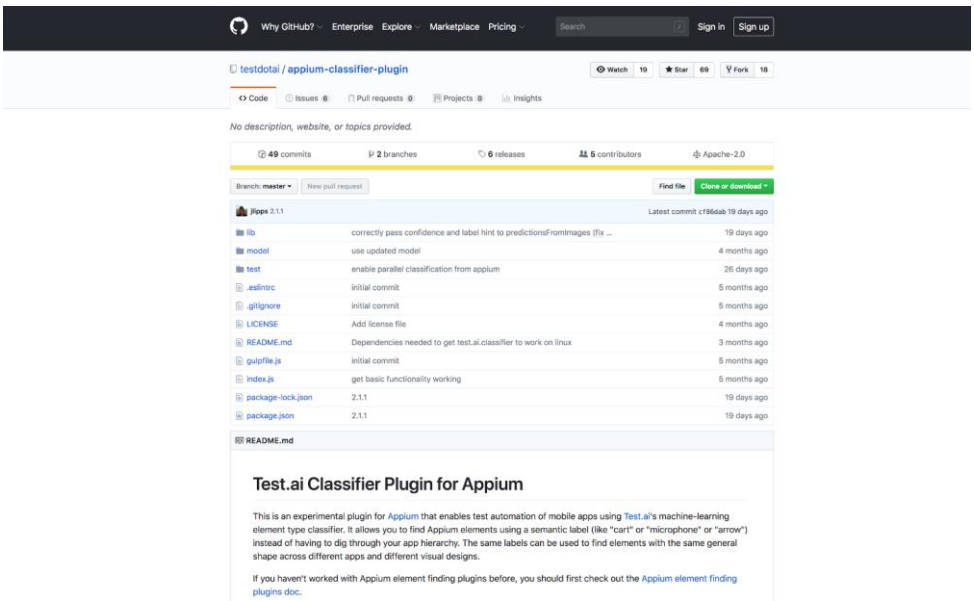
```
python run_model.py --image cart.png
```

66

Share with the World!



67



https://github.com/testdotai/appium-classifier-plugin

68

Performance Analysis

69

K-Means Clustering

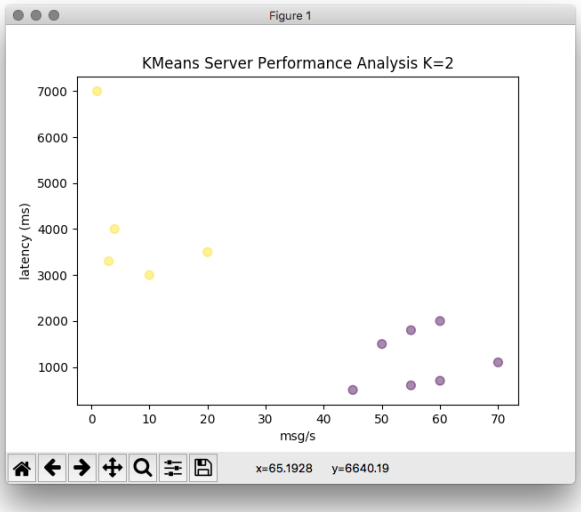
70

K Means

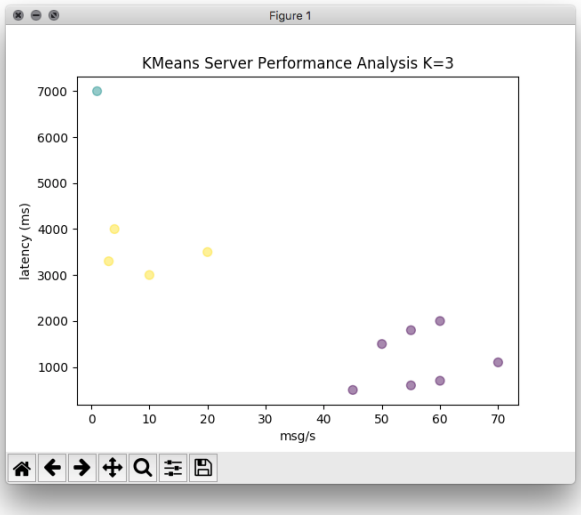
kmeans.py — Documents

```
#Jason Arbon
# Quick example of using Kmeans to identify servers
# with different performance characteristics

from pandas import DataFrame
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

Data = {'msg/s':   [ 60,    55,  50,   60,  70,  45,   55,   10,   20,   1,    4,    3],
        'latency': [ 2000, 1800, 1500, 700, 1100, 500, 600, 3000, 3500, 7000, 4000, 3300]
       }

def plot(Data, k):
  df = DataFrame(Data,columns=['msg/s','latency'])

  kmeans = KMeans(n_clusters=k).fit(df)
  plt.scatter(df['msg/s'], df['latency'], c= kmeans.labels_.astype(float), s=50, alpha=0.5)
  print(kmeans.labels_.astype(float))
  plt.title('KMeans Server Performance Analysis K=' + str(k))
  plt.ylabel('latency (ms)')
  plt.xlabel('msg/s')
  plt.show()

plot(Data, 2)
plot(Data, 3)
```


Auto Identify Differently Behaving Servers



Auto Identify Differently Behaving Servers



Questions

What data might be interesting to cluster on your team?

Perf
Stress
Crash logs
Test Results

STPCon: Testing Software with ML

Your Testing Problems

AI Applicable?

75

jason@test.ai

Appdiff, Inc. 40 Brannan St #100, San Francisco, CA 94107, United States

76

77

Test Generation