Usability of Web-based Trained Neural Network(NN) APIs: A Technical Report

Merve Selcuk-Simsek

March 20, 2018

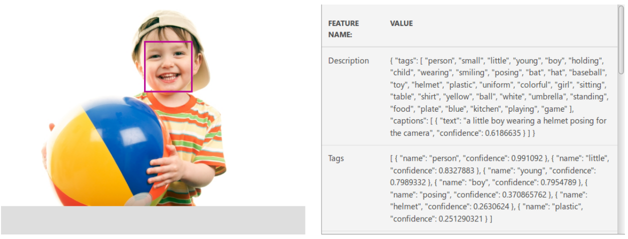
In this report I will present results of my examination obtained by using on-line trained neural network API(Application Programming Interface)s in computer vision concepts. These concepts cover topics of recognition of objects, faces, texts, and summarization of the image with descriptive tags. So far, ready to use web-based APIs which mostly belong to the leading companies in the sector such as Google(Section 2) and Microsoft(Section 1) are studied. As expected, they have some similarities and differences between them. Different parts of them will be explained in a detailed way along with examples in each section. Similarities could be listed as below:

- Object-face-text recognition (they could belong to the same or a separate division in the API),
- Information about image's general dynamics (e.g. gray-scale or RGB image, hue values of image etc.),
- Supplying recognition results with accuracy(or here confidence) values,
- Using the API either from a browser or within a programming language(PL).

1 Microsoft Azure

Depending on the expectation, Azure's service[1] might be or might not be sufficient as a computer vision API. As in all of the object recognition systems, this system as well gives the best results when the scene is less crowded, and objects are easier to identify. In Figure 1 and Figure 2 all the descriptive tags of the image given in the *tags* part with their accuracy values are correct. To put this in a more detailed way, in Figure 2b as stated in *tags* section, this image includes a person, green as a color, and it is seemingly a sample of indoor photography. However, repetitively, depending on what you would like to receive, this image could mean more than what is told in these tags which then makes the tags nothing but superficial. Same issue also stands for Figure 2a.

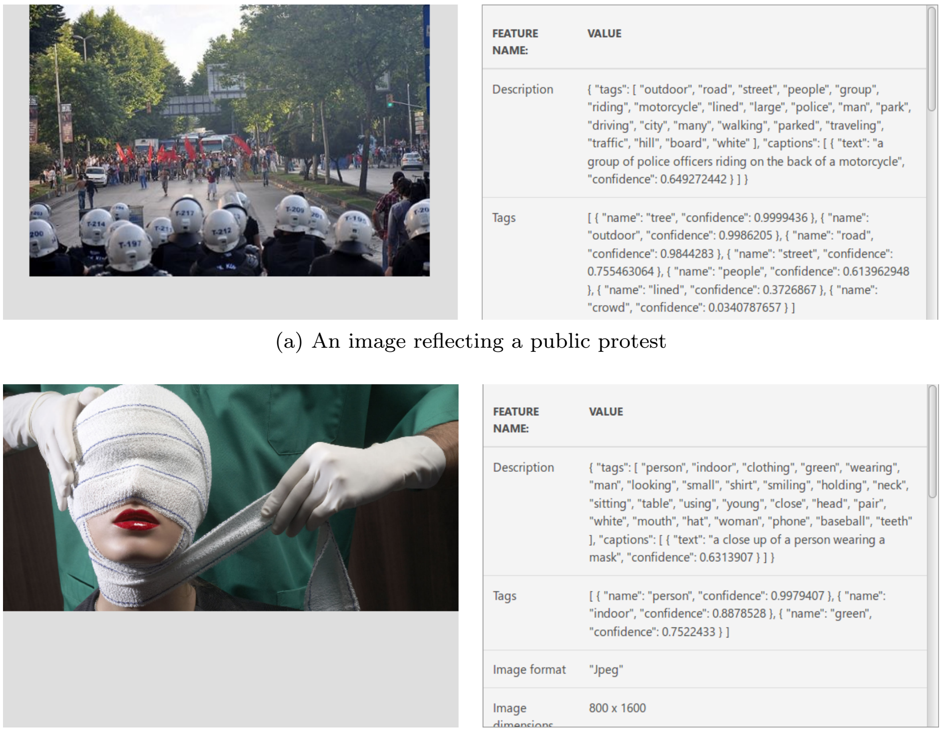
One extra feature that Azure has different than many others is a summarizing text (could be thought as a possible caption) of the images. This text takes place at the end of *Description* part of the each image which are output of the Computer Vision API. After looking into many images that four of them could be seen as Figure 1-3, I can say although not being always correct, producing a summary text could be defined as a good effort for Azure.



FEATURE NAME:	VALUE
Description	{ "tags": ["person", "small", "little", "young", "boy", "holding", "child", "wearing", "smiling", "posing", "bat", "hat", "baseball", "toy", "helmet", "plastic", "uniform", "colorful", "girl", "sitting", "table", "shirt", "yellow", "ball", "white", "umbrella", "standing", "food", "plate", "blue", "kitchen", "playing", "game"], "captions": [{ "text": "a little boy wearing a helmet posing for the camera", "confidence": 0.6186635 }] }
Tags	[{ "name": "person", "confidence": 0.991092 }, { "name": "little", "confidence": 0.8327883 }, { "name": "young", "confidence": 0.7989332 }, { "name": "boy", "confidence": 0.7954789 }, { "name": "posing", "confidence": 0.370865762 }, { "name": "helmet", "confidence": 0.2630624 }, { "name": "plastic", "confidence": 0.251290321 }]

Figure 1: An example image and its summary generated using Microsoft Azure's computer vision API

[1] https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision/



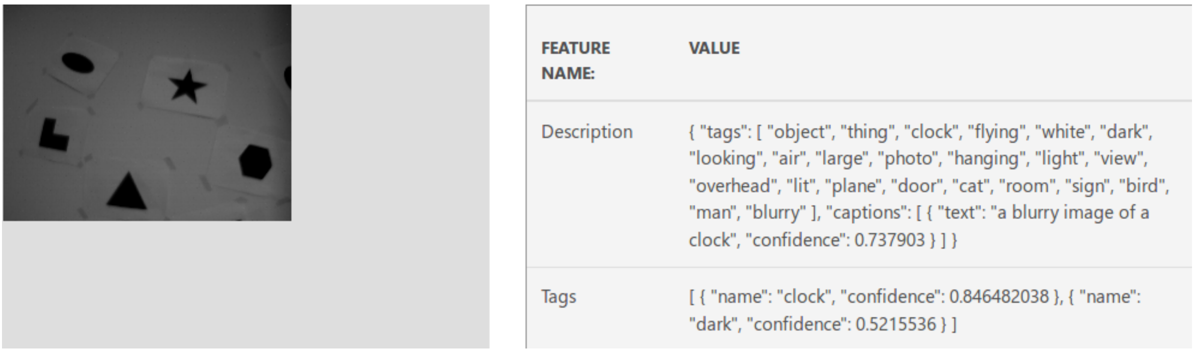
FEATURE NAME:	VALUE
Description	{ "tags": ["outdoor", "road", "street", "people", "group", "riding", "motorcycle", "lined", "large", "police", "man", "park", "driving", "city", "many", "walking", "parked", "traveling", "traffic", "hill", "board", "white"], "captions": [{ "text": "a group of police officers riding on the back of a motorcycle", "confidence": 0.649272442 }] }
Tags	[{ "name": "tree", "confidence": 0.9999436 }, { "name": "outdoor", "confidence": 0.9986205 }, { "name": "road", "confidence": 0.9844283 }, { "name": "street", "confidence": 0.755463064 }, { "name": "people", "confidence": 0.613962948 }, { "name": "lined", "confidence": 0.3726867 }, { "name": "crowd", "confidence": 0.0340787657 }]

(a) An image reflecting a public protest

FEATURE NAME:	VALUE
Description	{ "tags": ["person", "indoor", "clothing", "green", "wearing", "man", "looking", "small", "shirt", "smiling", "holding", "neck", "sitting", "table", "using", "young", "close", "head", "pair", "white", "mouth", "hat", "woman", "phone", "baseball", "teeth"], "captions": [{ "text": "a close up of a person wearing a mask", "confidence": 0.6313907 }] }
Tags	[{ "name": "person", "confidence": 0.9979407 }, { "name": "indoor", "confidence": 0.8878528 }, { "name": "green", "confidence": 0.7522433 }]
Image format	"Jpeg"
Image dimensions	800 x 1600

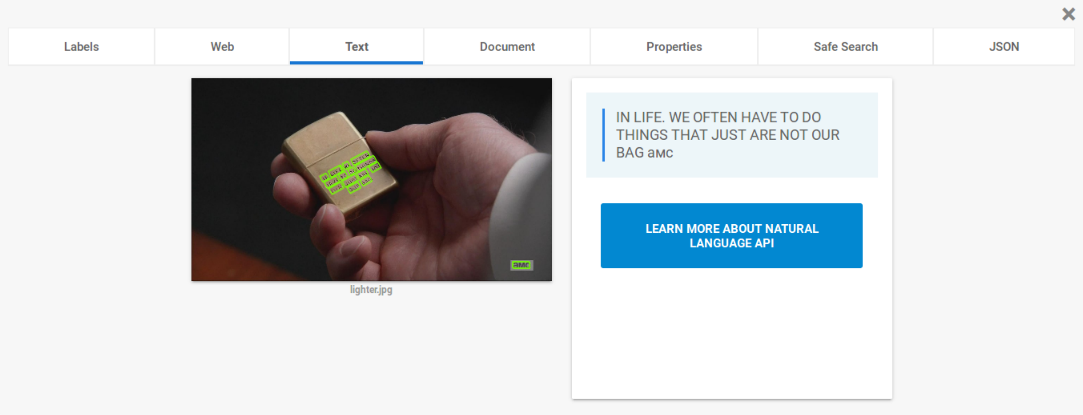
(b) A scene showing a woman's badges opening after a surgery

Figure 2: Two examples of image identification service of Azure in which description tags are correct, but summary text is wrong



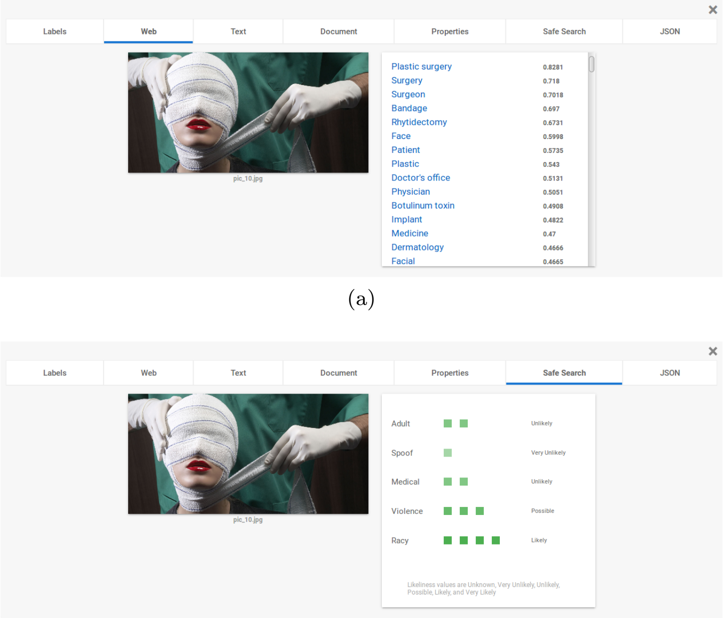
FEATURE NAME:	VALUE
Description	{ "tags": ["object", "thing", "clock", "flying", "white", "dark", "looking", "air", "large", "photo", "hanging", "light", "view", "overhead", "lit", "plane", "door", "cat", "room", "sign", "bird", "man", "blurry"], "captions": [{ "text": "a blurry image of a clock", "confidence": 0.737903 }] }
Tags	[{ "name": "clock", "confidence": 0.846482038 }, { "name": "dark", "confidence": 0.5215536 }]

Figure 3: A wrongly identified image from Azure



IN LIFE. WE OFTEN HAVE TO DO THINGS THAT JUST ARE NOT OUR BAG aMC

Figure 4: Text detection example using Google's API



Plastic surgery	0.8281
Surgery	0.718
Surgeon	0.7018
Bandage	0.697
Rhytidectomy	0.6731
Face	0.5998
Patient	0.5735
Plastic	0.543
Doctor's office	0.5131
Physician	0.5051
Botulinum toxin	0.4908
Implant	0.4822
Medicine	0.47
Dermatology	0.4666
Facial	0.4665

(a)

Adult	🟩🟩	Unlikely
Spoof	🟩	Very Unlikely
Medical	🟩🟩	Unlikely
Violence	🟩🟩🟩	Possible
Racy	🟩🟩🟩🟩	Likely

Likeliness values are Unknown, Very Unlikely, Unlikely, Possible, Likely, and Very Likely

(b)

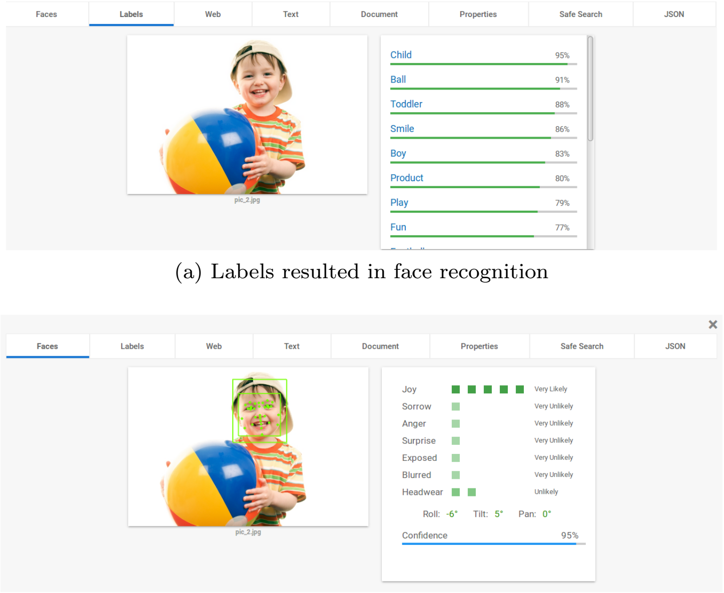
Figure 5: A relatively harder image to try on Google Cloud Vision API (a) *Web entity* search (b)*Safe search* example

2 Google Cloud Vision API

Google's Vision API[2] offers many solutions to the concept of understanding an image at the same time. The API is able to detect objects, texts (see Figure 4), faces (see Figure 6a), even facial mimics (see Figure 6b) in an image. Besides, it has other useful features such as *safe search* to detect and avoid possible unwanted entities in an image (see Figure 5b), and also, from my point of view the most importantly, it provides related web entities to the specified image which puts the user to the point of having more insight, and thus, understanding the image.

I wanted to try some unorthodox scenes as well as the regular ones on this API. In Figure 7 there are output obtained by feeding the API with a cartoon, and all the results, either belonging to the *labels* or the *web* section, are correct. The results in the *web* section supply every detail about the cartoon. It is not just for the cartoon example, but in *web* entities requested for every images (Figure 5a, 7b, 8b, 10b, 10d) we can obviously see that exploiting the fact of being the most used search engine, Google put excellent results forward with the "web entities" part to its image API. To illustrate, looking to the image in Figure 8b we can only say this is an arrest between a

[2] https://cloud.google.com/vision/



Child 95%
Ball 91%
Toddler 88%
Smile 86%
Boy 83%
Product 80%
Play 79%
Fun 77%

(a) Labels resulted in face recognition

Joy ■ ■ ■ ■ ■ Very Likely
Sorrow ■ Very Unlikely
Anger ■ Very Unlikely
Surprise ■ Very Unlikely
Exposed ■ Very Unlikely
Blurred ■ Very Unlikely
Headwear ■ ■ Unlikely

Roll: -6° Tilt: 5° Pan: 0°

Confidence 95%

(b) Detecting mimics in a face

Figure 6: Face recognition with Google Cloud Vision API

policeman and a man while other policemen were staring. However, it is indeed more than that, and we can easily learn about it from the web-entities part of the API.

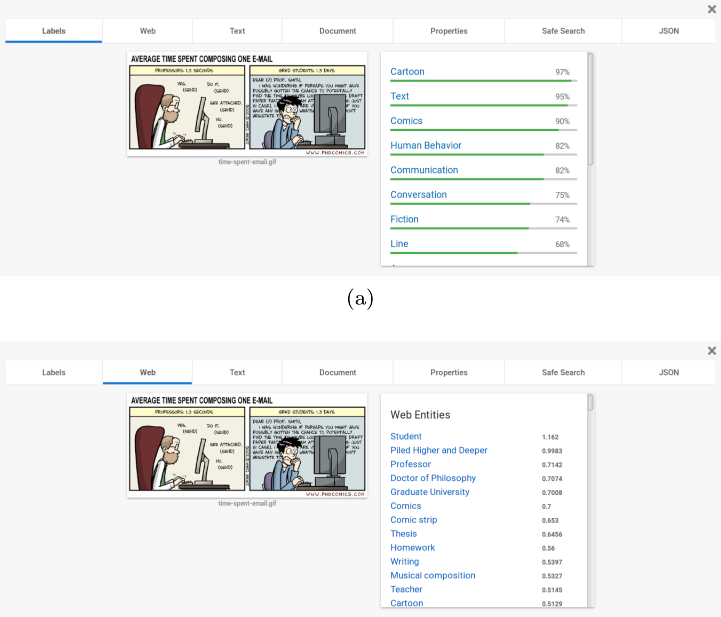
After having promising results, I wanted to take a risk and try images I produced consisting of events (i.e. only 2D points). In Figure 9 there are two frames which are indeed the same, but looks differently because of their way of being processed. On one hand, there is a frame taken by a conventional camera in Figure 9a. On the other hand, in Figure 9b, there is the same scene recorded by an event-based camera, and moreover, in order to detect objects in it processed using *k-means clustering* algorithm; so that we see 7 different objects painted in different colors and mapped onto a black canvas.

I fed the Google's vision API with the images in Figure 9, and although I didn't get entirely correct results, there were still encouraging results in my belief. Firstly, the scene itself could be hard to describe even for a human-being. Secondly, the results I got from Google's Vision API with the same image was definitely better than the one I got from Microsoft Azure's (see Figure 3). Finally, even though it is not enough on its own to describe the whole scene, among the tags in Figure 9, we see "star". After *star*, we also see some terms related to *star* as in *sky, night* which shows in this API not only they use object recognition techniques, but also "relative suggestions" in a way of searching for meaning as if trying to find missing pieces in a puzzle by trying the similar looking ones. However, the vital point here for me is to see a possible partnership between events and NNs.

Lastly, I run experiments and observed the results of the API with SLAM datasets. In Figure 10, two different datasets belonging to LSD-SLAM(Figure 10a-10b) and ORB-SLAM(Figure 10c-10d) are assessed on the API. Having better results on the Figure 10c and 10d, one thing surprised me on Figure 10a-10b was that nothing closer to a *computer* was detected on the scene. There is a monitor (possibly a PC), and a laptop on the desk; yet, none of them were tagged in results. Even if a computer-related item wasn't recognized, I was expecting at least something rectangular's like a book instead of a laptop being suggested.

I've got the most solid and enhanced results from that API. My overall review about Google's API would be as follows:

- It is successful to define image's main (not obvious even to human eye) identities such as a monochrome photography.
- It defines an image from many different perspectives. To illustrate, it looks not just for objects in an image, but also for texts, for people, and if there are people, then evaluates their facial expressions etc.
- Success of detecting all objects in the input depends to the scene itself, and definitely to



Cartoon	97%
Text	95%
Comics	90%
Human Behavior	82%
Communication	82%
Conversation	75%
Fiction	74%
Line	68%

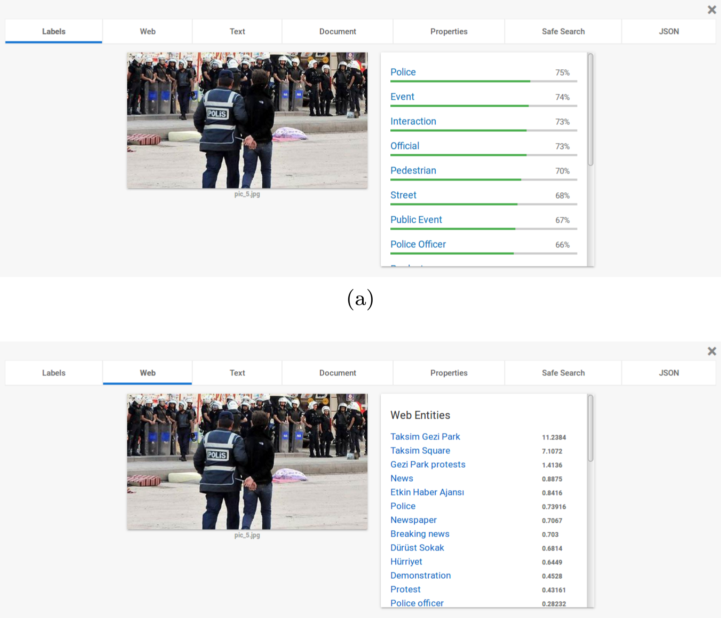
(a)

Web Entities

Student	1.162
Piled Higher and Deeper	0.9983
Professor	0.7142
Doctor of Philosophy	0.7074
Graduate University	0.7008
Comics	0.7
Comic strip	0.653
Thesis	0.6456
Homework	0.56
Writing	0.5397
Musical composition	0.5327
Teacher	0.5145
Cartoon	0.5129

(b)

Figure 7: Recognizing a cartoon with Google Cloud Vision API (a)Labels (b)Web entities



Police	75%
Event	74%
Interaction	73%
Official	73%
Pedestrian	70%
Street	68%
Public Event	67%
Police Officer	66%

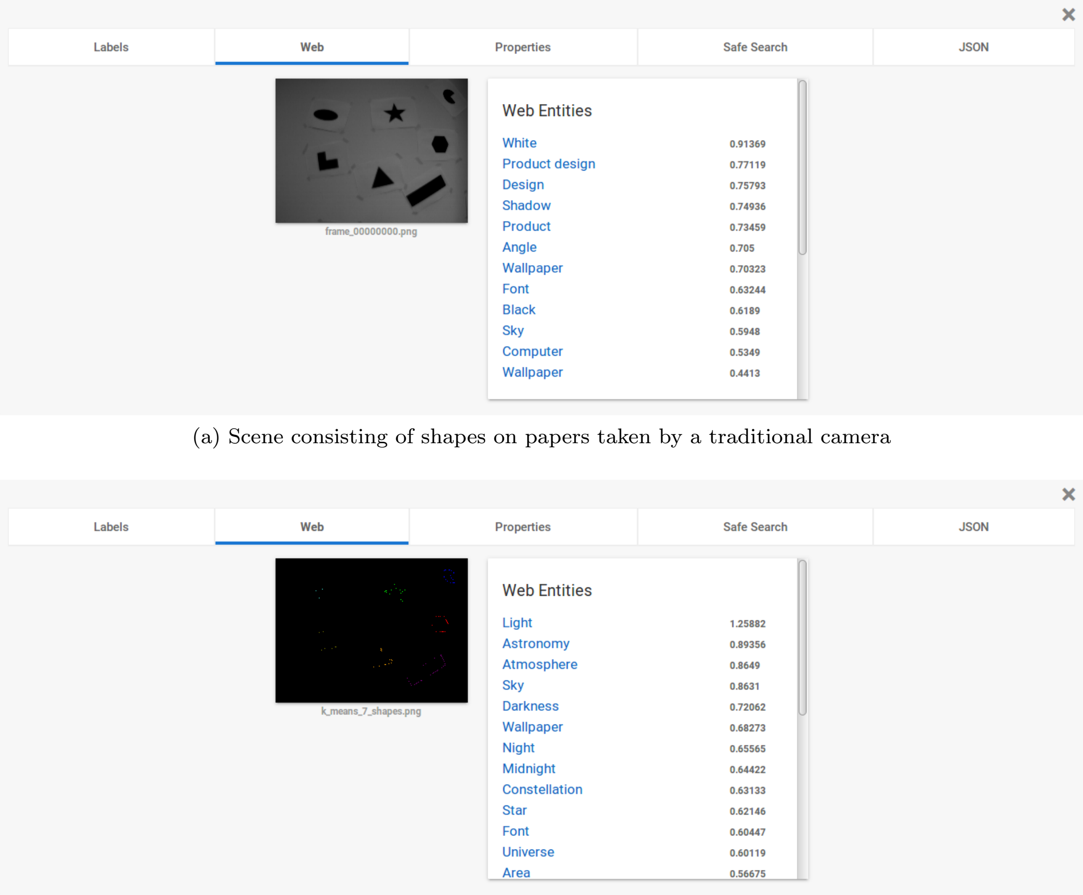
(a)

Web Entities

Taksim Gezi Park	11.2384
Taksim Square	7.1072
Gezi Park protests	1.4136
News	0.8875
Etkin Haber Ajansı	0.8416
Police	0.73916
Newspaper	0.7067
Breaking news	0.703
Dürüst Sokak	0.6814
Hürriyet	0.6449
Demonstration	0.4528
Protest	0.43161
Police officer	0.28232

(b)

Figure 8: Recognizing an arrest scene with Google Cloud Vision API (a)Labels (b)Web entities



Web Entities

White	0.91369
Product design	0.77119
Design	0.75793
Shadow	0.74936
Product	0.73459
Angle	0.705
Wallpaper	0.70323
Font	0.63244
Black	0.6189
Sky	0.5948
Computer	0.5349
Wallpaper	0.4413

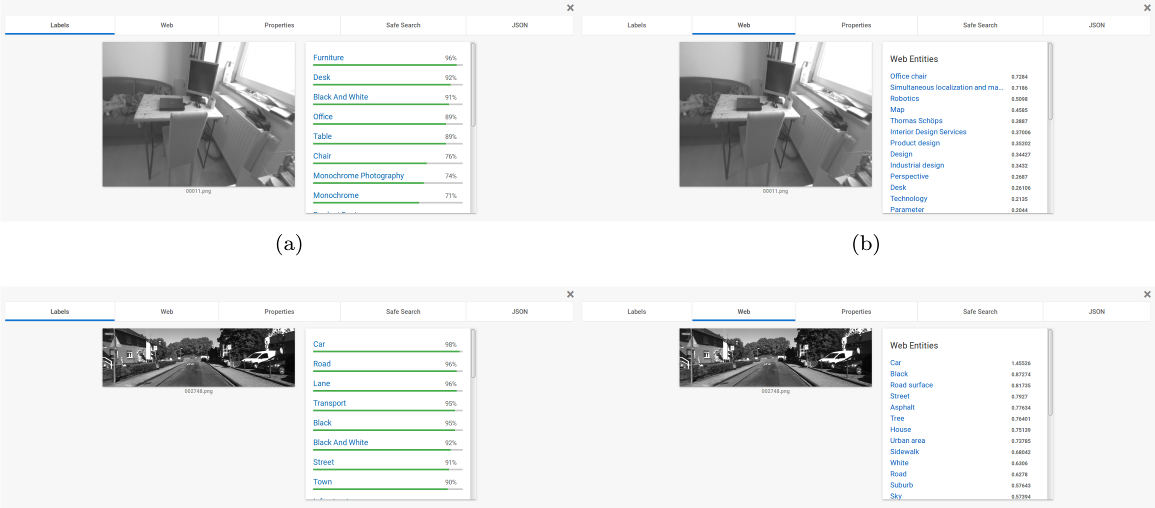
(a) Scene consisting of shapes on papers taken by a traditional camera

Web Entities

Light	1.25882
Astronomy	0.89356
Atmosphere	0.8649
Sky	0.8631
Darkness	0.72062
Wallpaper	0.68273
Night	0.65565
Midnight	0.64422
Constellation	0.63133
Star	0.62146
Font	0.60447
Universe	0.60119
Area	0.56675

(b) Same scene taken by an event-based camera and processed afterwards

Figure 9: Evaluation of the same scene which taken by different types of cameras



Furniture	96%
Desk	92%
Black And White	91%
Office	89%
Table	89%
Chair	76%
Monochrome Photography	74%
Monochrome	71%

(a)

Web Entities

Office chair	0.7284
Simultaneous localization and ma...	0.7186
Robotics	0.5098
Map	0.4585
Thomas Schöps	0.3887
Interior Design Services	0.37006
Product design	0.35202
Design	0.34427
Industrial design	0.3432
Perspective	0.2687
Desk	0.26106
Technology	0.2135
Parameter	0.2044

(b)

Car	98%
Road	96%
Lane	96%
Transport	95%
Black	95%
Black And White	92%
Street	91%
Town	90%

(c)

Web Entities

Car	1.45526
Black	0.87274
Road surface	0.81735
Street	0.7927
Asphalt	0.77634
Tree	0.76401
House	0.75139
Urban area	0.73785
Sidewalk	0.68042
White	0.6306
Road	0.6278
Suburb	0.57643
Sky	0.57394

(d)

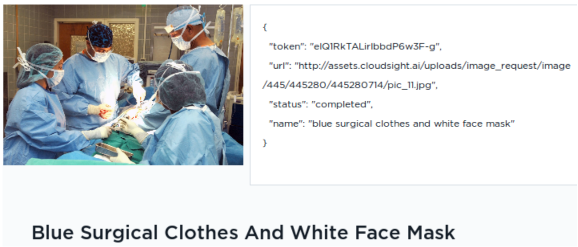
Figure 10: Scenes from SLAM Datasets with Google Cloud Vision API (a-b)Dataset used in LSD-SLAM (c-d)KITTI Dataset used in ORB-SLAM (a-c)labels (b-d)web entities

the neural network's training situation (e.g. if it wasn't fed enough with different type of drums, then it fails to detect them) as well; so, it is safe to say most of the time Cloud API is successful to detect objects.

- Text detection in images and mimic identification beside face detection are bonus features this API has.

3 Other On-line Vision APIs

There are some other API services alongside Microsoft's and Google's APIs such as IBM Watson[3], Amazon's Rekognition[4], CloudSight[5], and Clarifai[6]. So far, I tried CloudSight out of them, an example is presented in Figure 11. In my knowledge, the expertise of CloudSight is on generating headlines for images i.e. summarizing the scene in which it is the most successful API that I've seen so far.



```
{
  "token": "elQ1RkTALirlbbdP6w3F-g",
  "url": "http://assets.cloudsight.ai/uploads/image_request/image
/445/445280/445280714/pic_11.jpg",
  "status": "completed",
  "name": "blue surgical clothes and white face mask"
}
```

Blue Surgical Clothes And White Face Mask

Figure 11: An example from CloudSight's API

For more visual comparisons including all the web-based APIs mentioned above I find this site as a good source `https://goberoi.github.io/cloudy_vision/output/output.html`.

4 Conclusion

I report use cases of up-to-date on-line trained neural network APIs. Although I haven't attempt to use any of the systems via a PL, many of them have sufficiently informative documentation, and seem easily adaptable.

They have similarities and differences in use, and choosing the one to continue with depends definitely to the use case. Among the on-line interfaces, for general purposes, I vote on behalf Google Cloud due to its being very easy to use, showing all necessary information about an image at the same time (without having divisions such as "Face API", "Natural Language API" etc.), and giving the most accurate results on each image that I fed the system with.

I have some comments to make about supported PLs:

I had expected to see *python* for each one on the list, and I did.

I had expected to see *C++* for at least one of the APIs on the list, but I did not for any of them.

I hadn't expected to see *Java* for each one on the list, yet I did for all. However, considering the process with these APIs via a PL would be basically just a give-and-take between a server and a client, it should not be surprising indeed. Also, that could be thought as a new window opened to Java's computer vision adventure.

[3] https://www.ibm.com/watson/services/visual-recognition/

[4] https://aws.amazon.com/rekognition/

[5] https://cloudsight.ai/

[6] https://www.clarifai.com/

Credits

Figure 1, 6	https://ministry-to-children.com/
Figure 2a	https://www.haberler.com/
Figure 2b, 5	http://www.howesoundpp.com/
Figure 3, 9a	http://rpg.ifi.uzh.ch/software_datasets.html
Figure 4	https://www.reddit.com/r/madmen/
Figure 7	http://phdcomics.com/comics.php
Figure 8	http://sondakikahaberleri.info.tr/
Figure 10a, 10b	https://vision.in.tum.de/research/vslam/lsdslam
Figure 10c, 10d	http://www.cvlibs.net/datasets/kitti/index.php
Figure 11	http://dribrook.blogspot.co.uk/