evaluate_game

May 8, 2019

In [1]:
```
import os
os.environ["CUDA_DEVICE_ORDER"]= "PCI_BUS_ID"
os.environ["CUDA_VISIBLE_DEVICES"]= "3"

%matplotlib inline
from matplotlib import pyplot as plt

import numpy as np
import json, string
import cv2
from collections import Counter, defaultdict
from tabletext import to_text
```

In [2]:
```
from keras.models import load_model
from keras.models import Sequential, Model
from contextlib import redirect_stdout
```

```
Using TensorFlow backend.
```

In [4]:
```
# Question, Answer, Image Path, Object Coordinates
qai_data = np.load("../preprocessed_data/scratch/gw_test_Q_A_ImgPath.npy")
# Image, Object, Spatial, Category, QA, Status
vec_data = np.load("../preprocessed_data/scratch/gw_test_vec.npy")
# History data
his_data = np.load("../preprocessed_data/scratch/hist_test.npy")
# Category Name with id
category_list = json.load(open("../preprocessed_data/scratch/category_list"))
# Category
cat_data = np.load("../preprocessed_data/scratch/gw_test_cat_id.npy")
```

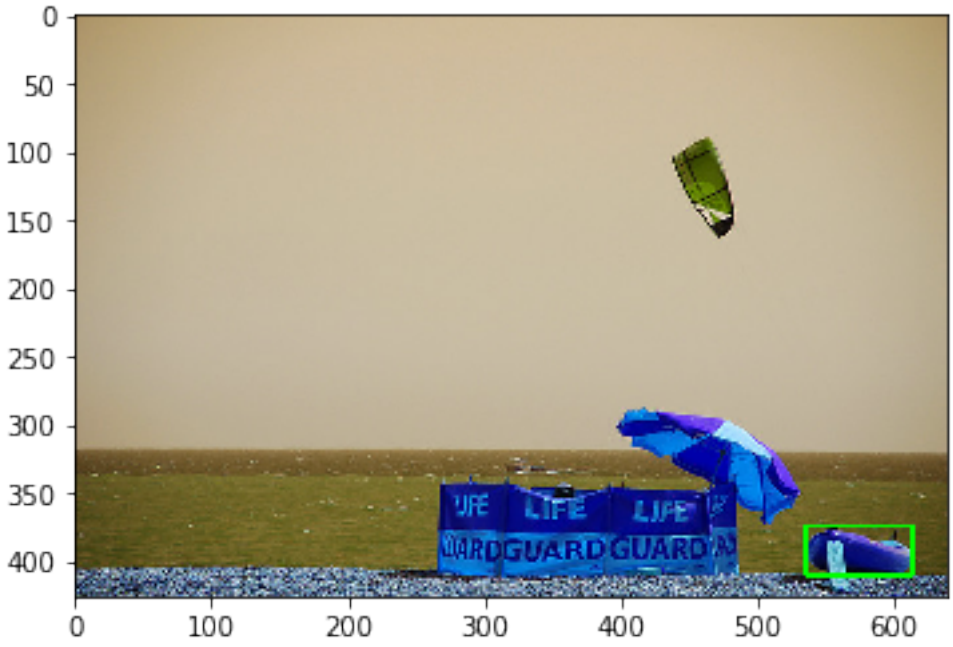
In [5]:
```
question = [i[0] for i in qai_data] # List of questions in the game
answer = [i[1] for i in qai_data] # List of answers in the game
img_path = [i[2] for i in qai_data] # Image path to read the image
obj_coor = [i[3] for i in qai_data] # Object co-ordinates to draw bounding box
spatial = [i[2] for i in vec_data] # Spatial co-ordinates of the object
status = [i[5] for i in vec_data] # Status of the game
#cat_data = [i[3] for i in vec_data] # Category id of the object
```

```
In [7]: def spatial_preprocess(i):
            x_width = i[2]
            y_height = i[3]
            x_left = i[0]
            x_right = i[0] + i[2]
            y_upper = 224 - i[1]
            y_lower = y_upper - y_height
            x_center = x_left + 0.5*x_width
            y_center = y_lower + 0.5*y_height
            #print(x_left,x_right,y_upper,y_lower,x_width,y_height,x_center,y_center)
            x_left = round((1.*x_left / 224) * 2 - 1,4)
            x_right = round((1.*x_right / 224) * 2 - 1,4)
            x_center = round((1.*x_center / 224) * 2 - 1,4)

            y_lower = round((1.*y_lower / 224) * 2 - 1,4)
            y_upper = round((1.*y_upper / 224) * 2 - 1,4)
            y_center = round((1.*y_center / 224) * 2 - 1,4)

            x_width = round((1.*x_width / 224) * 2,4)
            y_height = round((1.*y_height / 224) * 2,4)
            return [x_left,x_right,y_upper,y_lower,x_width,y_height,x_center,y_center]

In [8]: spatial = [spatial_preprocess(i) for i in spatial]

In [10]: # Splitting the history data w.r.t the games
         his_game = []
         xx = 0
         for i in range(len(question)):
             yy = xx
             xx = xx + len(question[i])
             his_game.append(his_data[yy:xx])

In [11]: # Load the model name
         # Input are q,c,i,s,o,h
         # Format ex: q+c+h.h5
         # Bilstm model need an "infix" in the model ex: q+c+h"_bilstm".h5
         model = load_model("../preprocessed_data/scratch/Results/q+c+s.h5")
         model1 = load_model("../preprocessed_data/scratch/Results/q+c+s+h.h5")
         model2 = load_model("../preprocessed_data/scratch/Results/q+c+s+h_bilstm.h5")

In [12]: # Function takes an integer that is the index of the game in test dataset
         # Returns the image, questions in the game, answers in the game, predicted answers fr
         # category, status, object co-ordinates
         def evaluate_function(game_no):
             list_question = question[game_no]
             img_file = img_path[game_no].replace('coco', '/usr/local/courses/lt2318/data/msco
             img = cv2.imread(img_file)
             spatial_value = spatial[game_no]
             category = cat_data[game_no]
```

```
            game_status = status[game_no]
            history = his_game[game_no]
            obj_location = obj_coor[game_no]
            his_question = [i[:-1] for i in history]
            las_question = [i[-1][:-1] for i in history]
            las_answer = [i[-1][-1] for i in history]
            results = model.predict([np.array(las_question), np.array([category]*len(las_ques
            results1 = model1.predict([np.array(las_question), np.array([category]*len(las_qu
            results2 = model2.predict([np.array(las_question), np.array([category]*len(las_qu
            WoH_classes = results.argmax(axis=-1)
            WH_classes = results1.argmax(axis=-1)
            WH_Bilstm = results2.argmax(axis=-1)
            return img, list_question, las_answer, WoH_classes, WH_classes, WH_Bilstm, catego
```

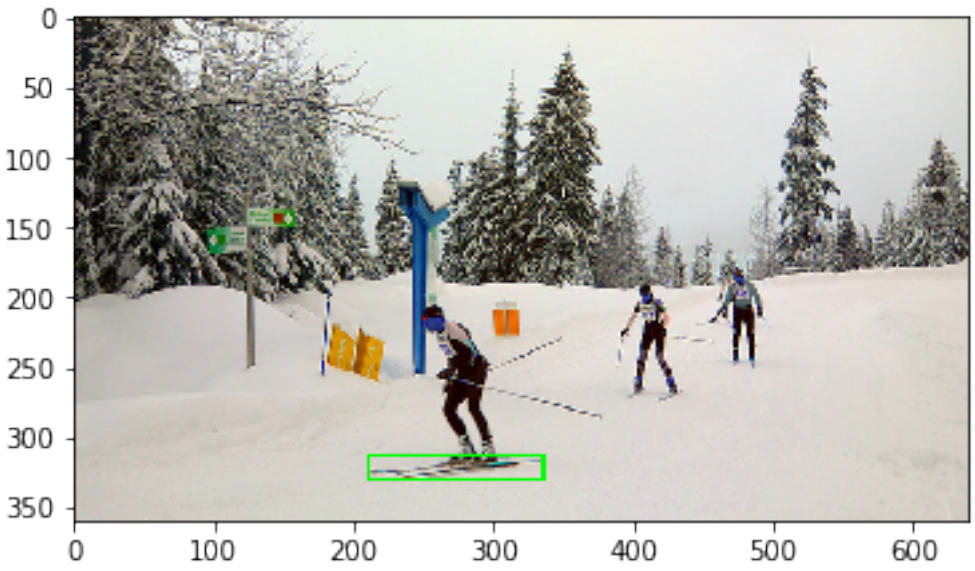
```
In [16]: for i in list_success[:5]:
            img, questions, true_answers, pred_answers_WoH, pred_answers_WH, pred_answers_WH_
            x,y,w,h = object_loc
            print("Category : %s, Status : %s" % (category_list[str(category)], game_status))
            data = [["Question", "Answer", "Pred_woH", "Pred_wH", "Pred_wH_Bi"]]
            for i in range(len(questions)):
                data.append([questions[i], true_answers[i], pred_answers_WoH[i], pred_answers_
            print(to_text(data))
            img = cv2.rectangle(img, (int(x),int(y)), (int(w),int(h)),(0,255,0), 2)
            plt.imshow(img)
            plt.show()
```

```
Category : boat, Status : success

 Question            Answer  Pred_woH  Pred_wH  Pred_wH_Bi

 is it in the sky         1         1        1           1

 is it the umbrella       1         1        1           1

 is it the ocean          1         1        1           1

 is it the lifeboat       0         0        0           0
```



Category : skis, Status : success

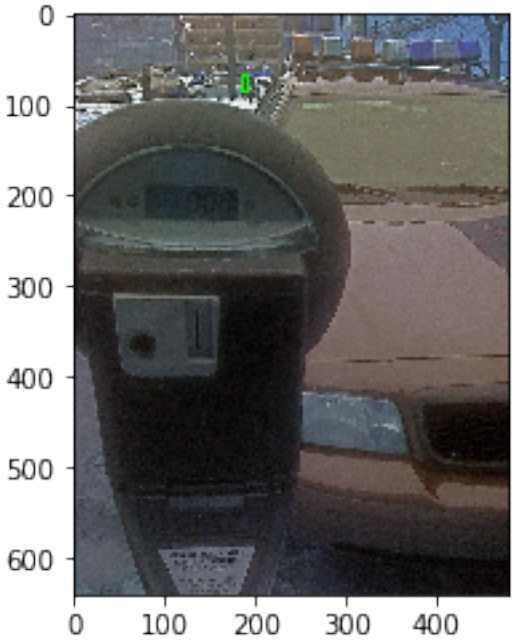
Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it a skier	1	1	1	1
is it blue	1	1	1	1
it it a pole	1	1	1	1
is it a tree	1	1	1	1
it is a ski	0	0	0	0
is it the front most skis	0	1	1	1



Category : parking meter, Status : success

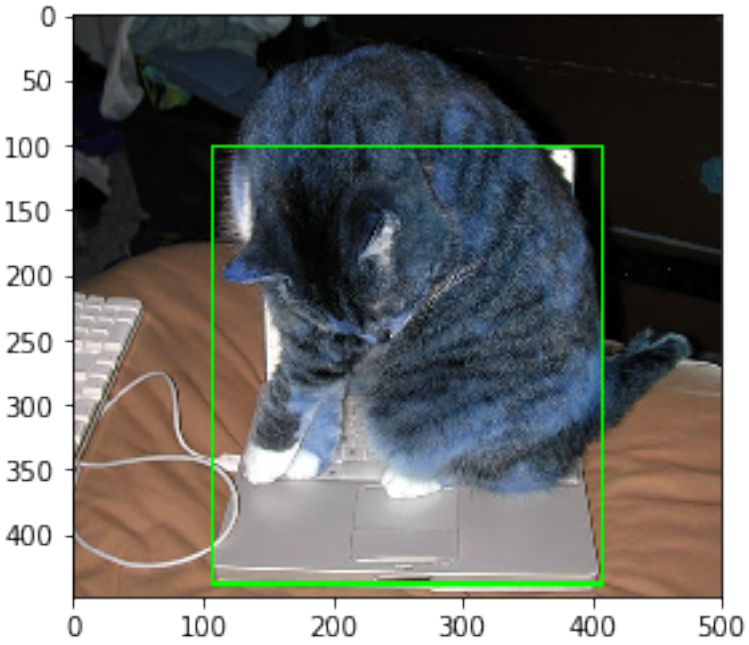
Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it a car	1	1	1	1
is it black	1	1	1	0
is it blue	1	1	1	1
is it red	1	0	1	1
is it a tree	1	1	1	1
is it white	1	1	1	1
is it silver	1	1	1	1
is it a building	1	1	1	1
is it a person	1	1	1	1
is it a number	1	1	1	1
is it a word	1	1	1	1
is it orange	1	1	1	1

is it in the foreground	1	0	0	0
is it in the background	0	1	1	1
is it a pole	1	1	1	1
is it a structure	1	1	1	1
is it a window	1	1	1	1
is it a parking meter	0	0	0	0



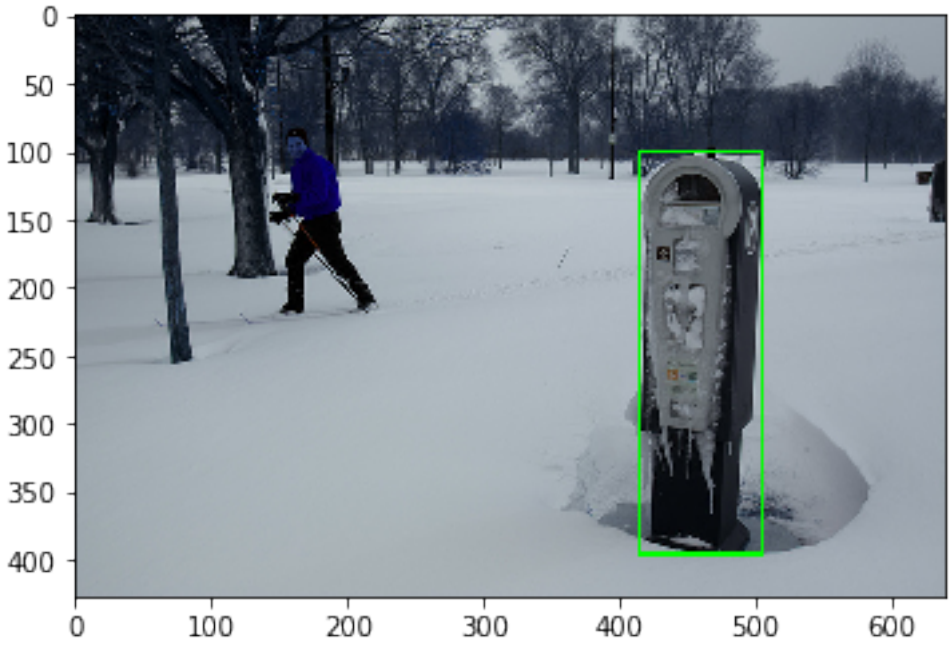
Category : laptop, Status : success

Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it an animal	1	1	1	1
is it a device	0	0	0	0
is it silver in color	0	1	1	1



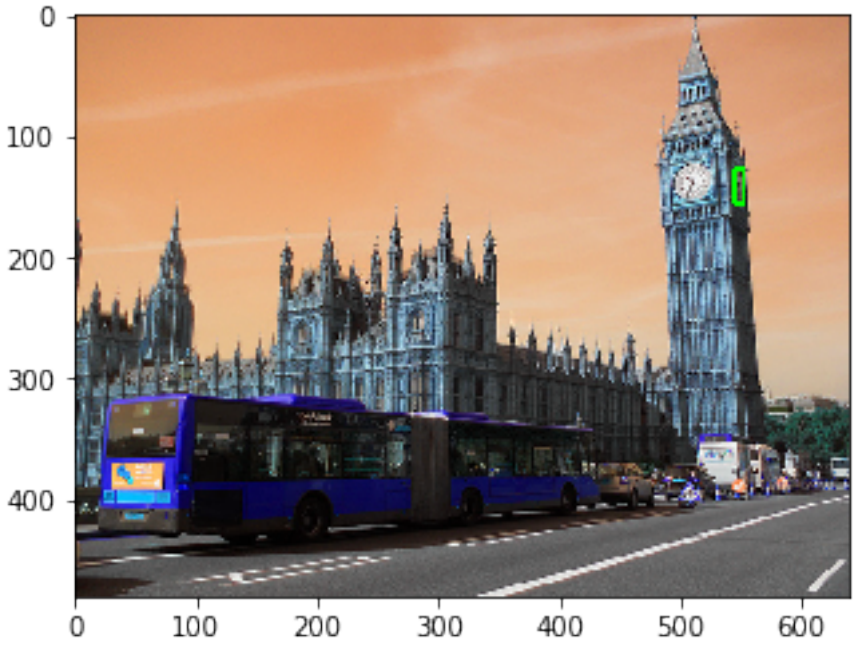
Category : parking meter, Status : success

Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it natural	1	1	1	1
is it close to the camera	0	0	0	0
is there a letter p on the side	0	0	1	1




```
In [20]: for i in list_failure[25:30]:
             img, questions, true_answers, pred_answers_WoH, pred_answers_WH, pred_answers_WH_
             x,y,w,h = object_loc
             print("Category : %s, Status : %s" % (category_list[str(category)], game_status))
             data = [["Question", "Answer", "Pred_woH", "Pred_wH", "Pred_wH_Bi"]]
             for i in range(len(questions)):
                 data.append([questions[i], true_answers[i], pred_answers_WoH[i], pred_answers_
             print(to_text(data))
             img = cv2.rectangle(img, (int(x),int(y)), (int(w),int(h)),(0,255,0), 2)
             plt.imshow(img)
             plt.show()
```

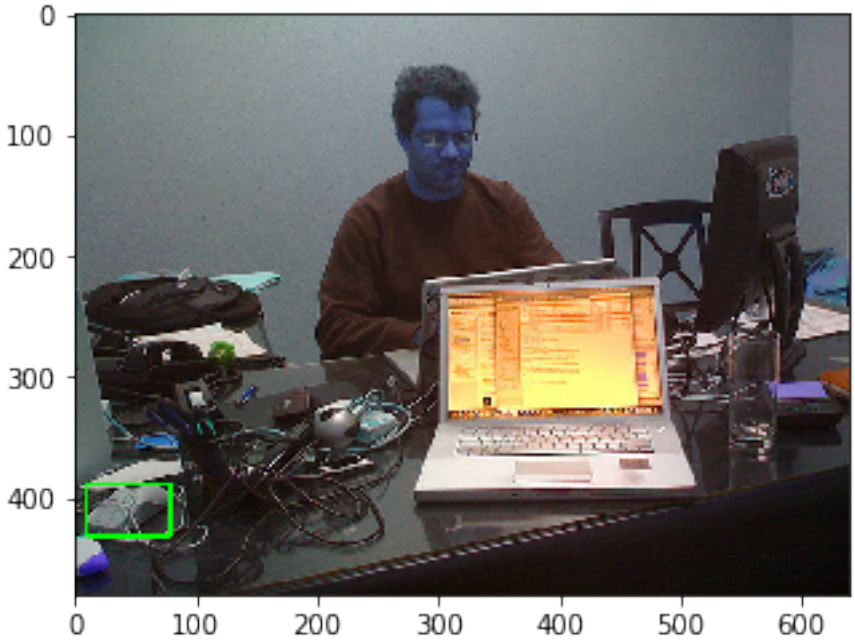
```
Category : clock, Status : failure

 Question              Answer  Pred_woH  Pred_wH  Pred_wH_Bi

 is it a clock              0         1        0           1

 is it a tower              0         1        1           1

 is it a clock tower        0         0        0           0
```



Category : mouse, Status : failure

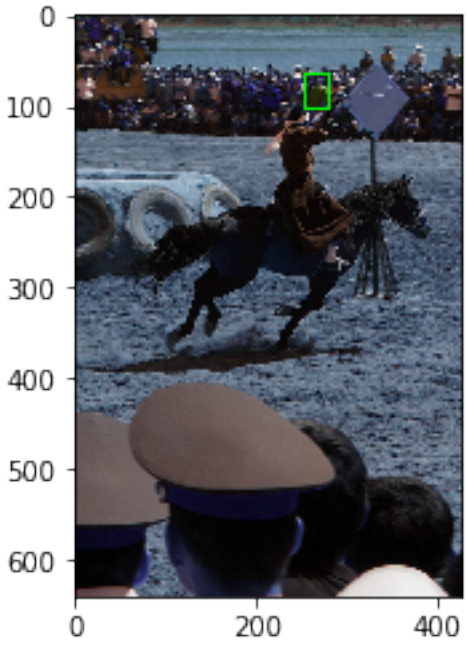
Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it electronic	0	0	0	0
is there something on the screen	1	1	1	1
is a person staring at it	1	0	0	0
is it silver	2	1	0	0
is it black	1	0	0	0
is it white	1	0	1	0
is it blue	1	1	1	1
does it have buttons	0	0	0	0
is there red square on top of it	1	0	1	0
does it have a keyboard	1	0	1	0
can you use it on a computer	0	0	0	0

is it round	1	1	1	0



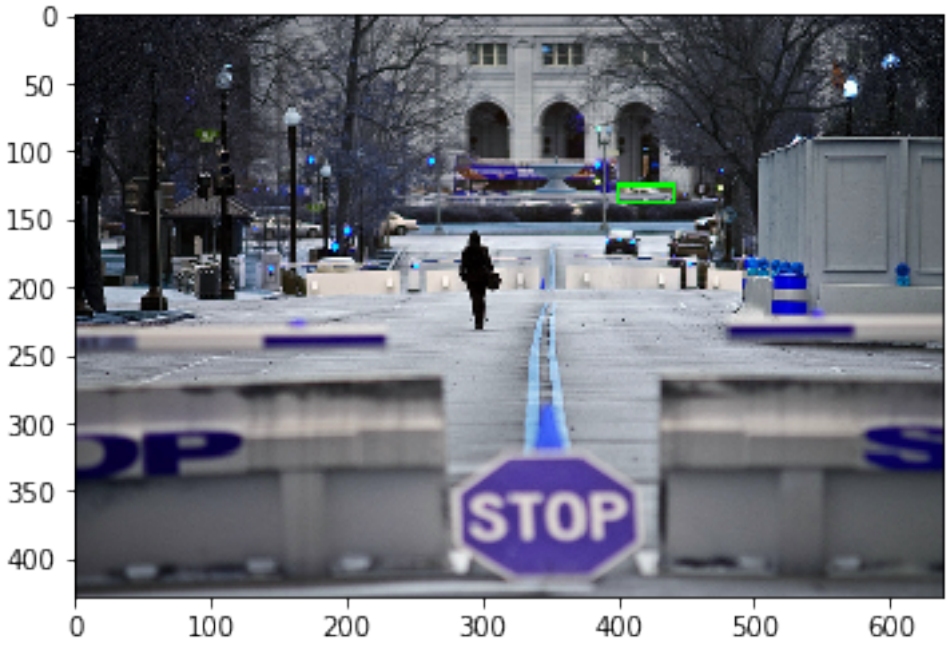
Category : person, Status : failure

Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it a person	0	0	0	0
are they riding a horse	1	1	0	0
are they facing away from the camera	1	0	1	1
are they standing up	0	0	0	0
are they wearing white	1	1	0	0
is this person standing between two people	0	1	1	1



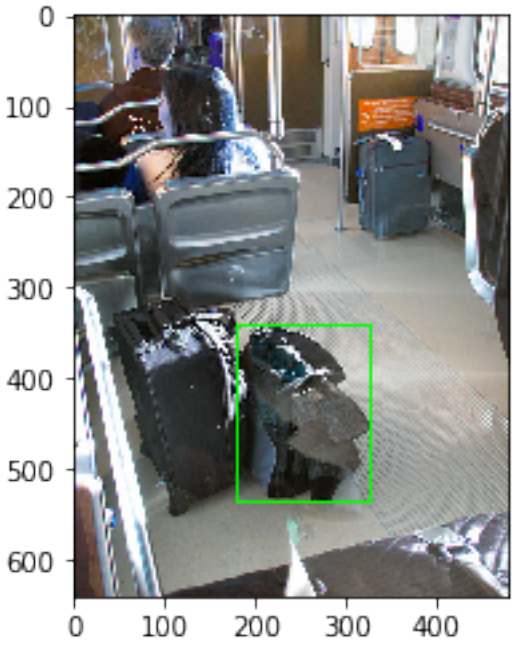
Category : car, Status : failure

Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it orange	1	1	1	1
is it a sign	1	1	1	1
is it a person	1	1	1	1
is it a vehicle	0	0	0	0
is it white	1	0	0	0
is it long	1	1	1	1
is it black	1	1	0	0
is the front part behind a tree	1	0	0	0
is it behind a light pole	1	1	1	1
is it red	1	1	0	0



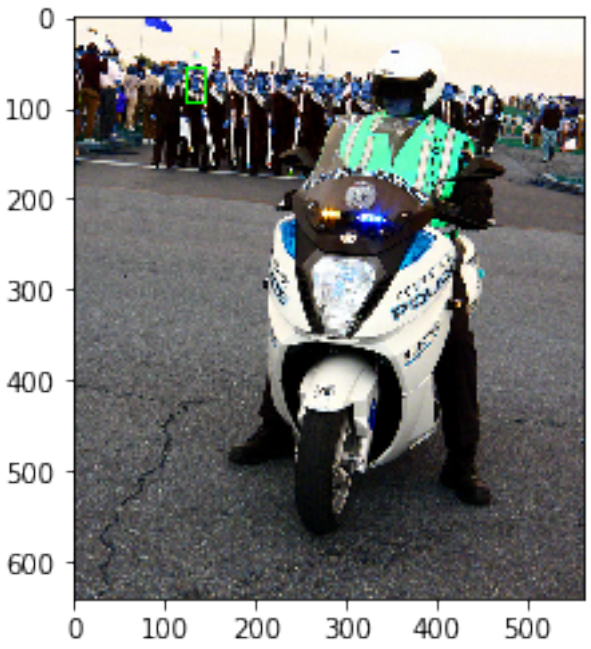
Category : backpack, Status : failure

Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it a person	1	1	1	1
is it a piece of luggage	0	0	0	0
is it black	1	0	1	1




```
In [27]: for i in list_incomplete[45:50]:
             img, questions, true_answers, pred_answers_WoH, pred_answers_WH, pred_answers_WH_
             x,y,w,h = object_loc
             print("Category : %s, Status : %s" % (category_list[str(category)], game_status))
             data = [["Question", "Answer", "Pred_woH", "Pred_wH", "Pred_wH_Bi"]]
             for i in range(len(questions)):
                 data.append([questions[i], true_answers[i], pred_answers_WoH[i], pred_answers_
             print(to_text(data))
             img = cv2.rectangle(img, (int(x),int(y)), (int(w),int(h)),(0,255,0), 2)
             plt.imshow(img)
             plt.show()
```

```
Category : person, Status : incomplete

 Question               Answer  Pred_woH  Pred_wH  Pred_wH_Bi

 is it a human               0         0        0           0

 is he riding the bike       1         0        0           0
```

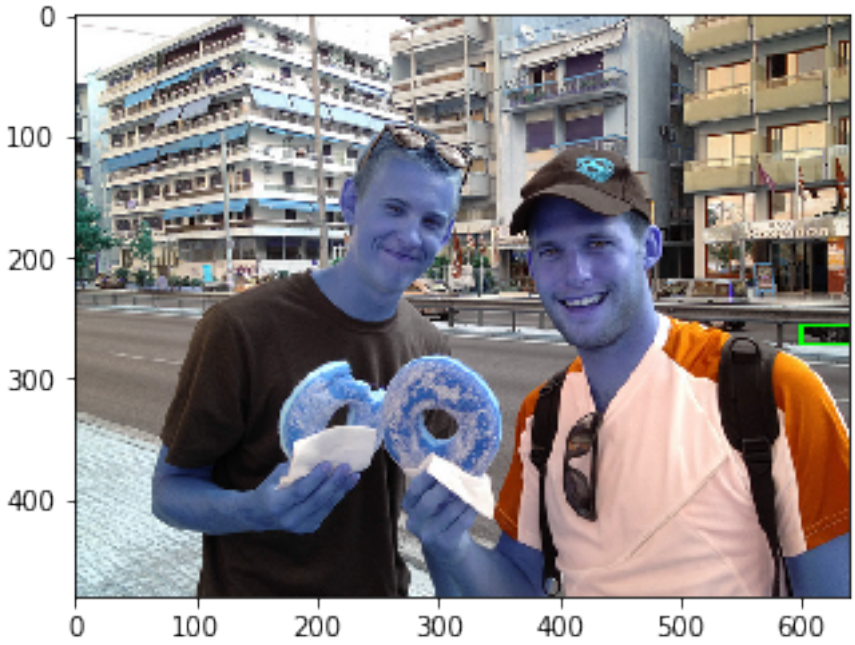


Category : person, Status : incomplete

Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it a person	0	0	0	0
is it a man	2	0	1	1
is it a woman	2	1	2	2

Category : car, Status : incomplete

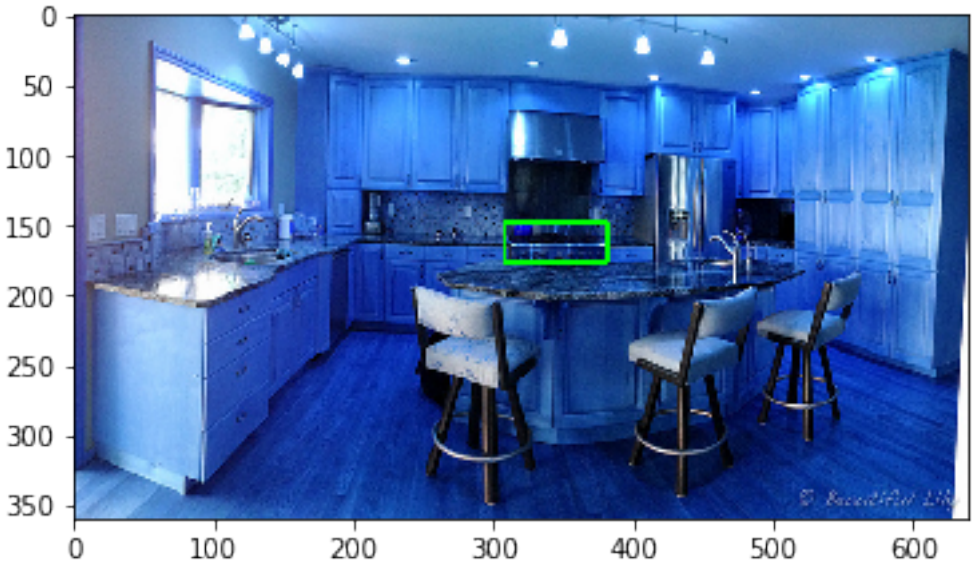
Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it a person	1	1	1	1
is it food	1	1	1	1
is it something on a person	1	1	1	1
is it a building	1	1	1	1
is it something attached to a building	1	1	1	1
is it a car	0	0	0	0
is the entire car in view	1	1	1	1



Category : oven, Status : incomplete

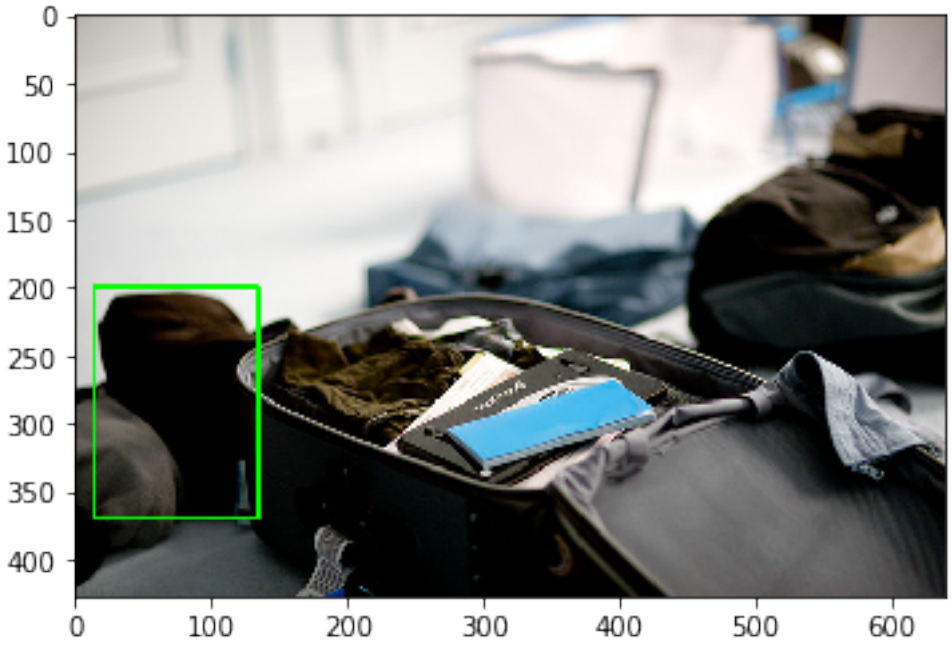
Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it furniture	1	1	1	1
is it near the sink	1	1	1	1
is it the green bottle	1	1	1	1
is it on the stove	1	1	1	1
is it below the cabinets	0	1	1	1
is it like a kettle	1	1	1	1
is it bread box	1	1	1	1
is it coffee maker	1	1	1	1
is it bottle	1	1	1	1
is it made of metal	0	0	0	1
is it fridge	1	1	1	1

is it hood over stove	1	1	1	1
is it taps	1	1	1	1
is it two metallic things on counter	1	1	1	1



Category : suitcase, Status : incomplete

Question	Answer	Pred_woH	Pred_wH	Pred_wH_Bi
is it a bag	0	0	0	0
the whole bag	1	0	0	0
is it in the front of the picture	1	1	1	1
is it the bag to the right of the picture	1	0	1	1
is it the bag at the back	1	0	1	1
is it the bag to the left of the picture	1	1	1	1




```
In [21]: list_failure = [i for i,j in enumerate(status) if j == "incomplete"]
```