Challenge 7: Deeper into Object Detection

Background

The Adventure Works data science team wants to experiment with creating a custom object detection model using a convolutional neural network (CNN). This custom model should perform the same object detection task as the model previously built with the *Computer Vision* service, detecting whether or not each person in an image is protected by a helmet.

Prerequisites

- An environment for sharing code and working in Jupyter.
- An installation of a deep learning framework with which to train an object detection model.
- The safety dataset. If you have not already downloaded this, a list of image URLs for training and testing can be found here. You can run the following code in a Jupyter notebook cell to download the images:

```
import os
import shutil
import requests
from io import BytesIO
from PIL import Image

# Create an empty folder
folder = 'safety_images'
if os.path.exists(folder):
    shutil.rmtree(folder)
os.makedirs(folder)

# Get the list of image URLs
!curl https://challenge.blob.core.windows.net/challengefiles/summit_post_urls_selected.txt -o urls.txt
urls = open("urls.txt", "r")

# Download each image
for url in urls.readlines():
    url = url.rstrip()
    filename = url.split('/')[-1]
    response = requests.get(url)
    img = Image.open(BytesIO(response.content))
    saveAs = os.path.join(folder, filename)
    print("writing " + saveAs)
    img.save(saveAs, 'JPEG')
```

Challenge

1. Using the deep learning framework of your choice, create an object detection solution. This model should be able to detect and create a bounding box around each helmet present in an image.
2. Test your model using an image that was not included in the training dataset, displaying the detected objects, their classes, and their bounding boxes.

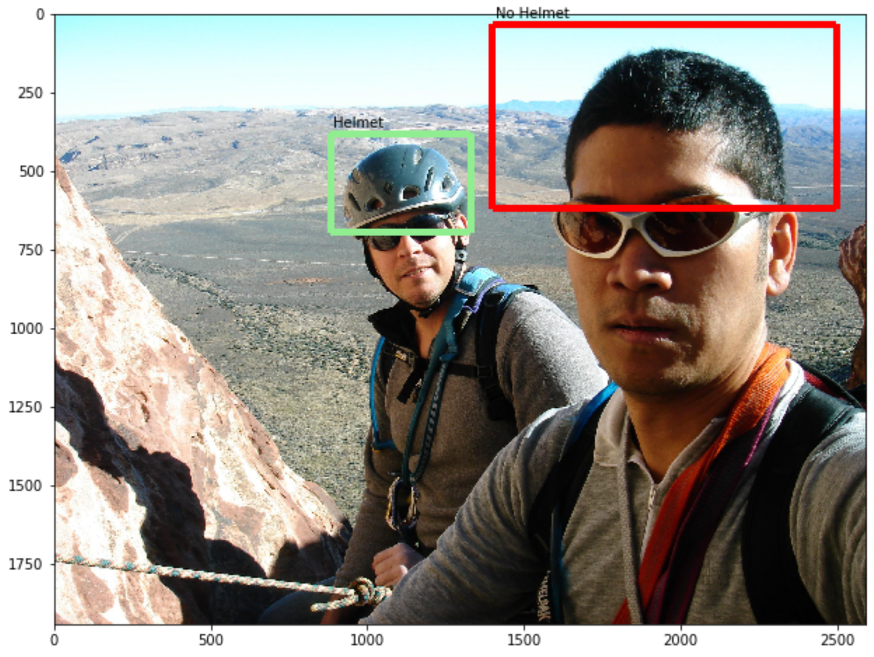
Hints

- The VoTT utility can output tags for training in formats suitable for the commonly used deep learning frameworks.
- There are many existing models for object detection, such a *YOLO*, *FAST R-CNN* and *Faster R-CNN* that you can use as the base model for *transfer learning*.

Success Criteria

- Your object detection model must achieve a Mean Average Precision (MAP) of **60%** or higher.

- You must write code that uses the model to get predictions for the classes and locations of objects in a test image, and plots the image overlaid with annotated bounding boxes for the predicted classes, like this:



Helmet Detection

References

Concepts

- What is object detection?
- What is MAP?

Tools and Frameworks

- Visual Object Tagging Tool (VOTT)
- Faster R-CNN in PyTorch
- The Tensorflow Object Detection API
- Object Detection with CNTK