💡 Inspiration:

Purpose: to object detection in images by using **YOLOv8**.

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. When it comes to object detection, popular detection frameworks are

- YOLO
- SSD
- Faster R-CNN

What is YOLO exactly?

What is YOLO exactly?

YOLO (You Only Look Once) is a method / way to do object detection. It is the algorithm /strategy behind how the code is going to detect objects in the image.

Earlier detection frameworks, looked at different parts of the image multiple times at different scales and repurposed image classification technique to detect objects. This approach is slow and inefficient.

YOLO takes entirely different approach. It looks at the entire image only once and goes through the network once and detects objects. Hence the name. It is very fast. That's the reason it has got so popular.

Import Libraries

In [124]:

```
import numpy as np
import pandas as pd
import cv2

from sklearn.utils import shuffle
from matplotlib.patches import Rectangle
import matplotlib.pyplot as plt

import warnings

warnings.simplefilter('ignore')
```

Load data

In [125]:

```
df = pd.read_csv('../input/self-driving-cars/labels_train.csv')
df = shuffle(df)
df.head()
```

Out[125]:

```
                          frame  xmin  xmax  ymin  ymax  class_id
83011  1479501521678379162.jpg   367   459   141   174         1
88841  1479502234720350336.jpg   130   145   142   153         1
91273  1479502425239194828.jpg   191   199   140   147         1
37755  1478896893517025816.jpg   134   154   156   171         1
18700  1478732200521726268.jpg   143   152    94   109         5
```

- This dataset is mostly a practice and training aspect for the subject of YOLO. The data is labeled with 5 classes.
- classic_id labels: 'car', 'truck', 'pedestrian', 'bicyclist', 'light'

In [126]:

```
classes = df.class_id.unique()
print(classes)
```

```
[1 5 4 2 3]
```

In [127]:

```
labels = { 1:'car', 2:'truck', 3:'person', 4:'bicycle', 5:'traffic light'}
```

Show some labeled images

In [128]:

```python
# Get path images and boxes (x,y) for each class_id
boxes = {}
images = {}

base_path = '../input/self-driving-cars/images/'

for class_id in classes:
    first_row = df[df['class_id'] == class_id].iloc[0]

    images[class_id] = cv2.imread(base_path + first_row['frame'])
    boxes[class_id] = [first_row['xmin'],first_row['xmax'],first_row['ymin
```

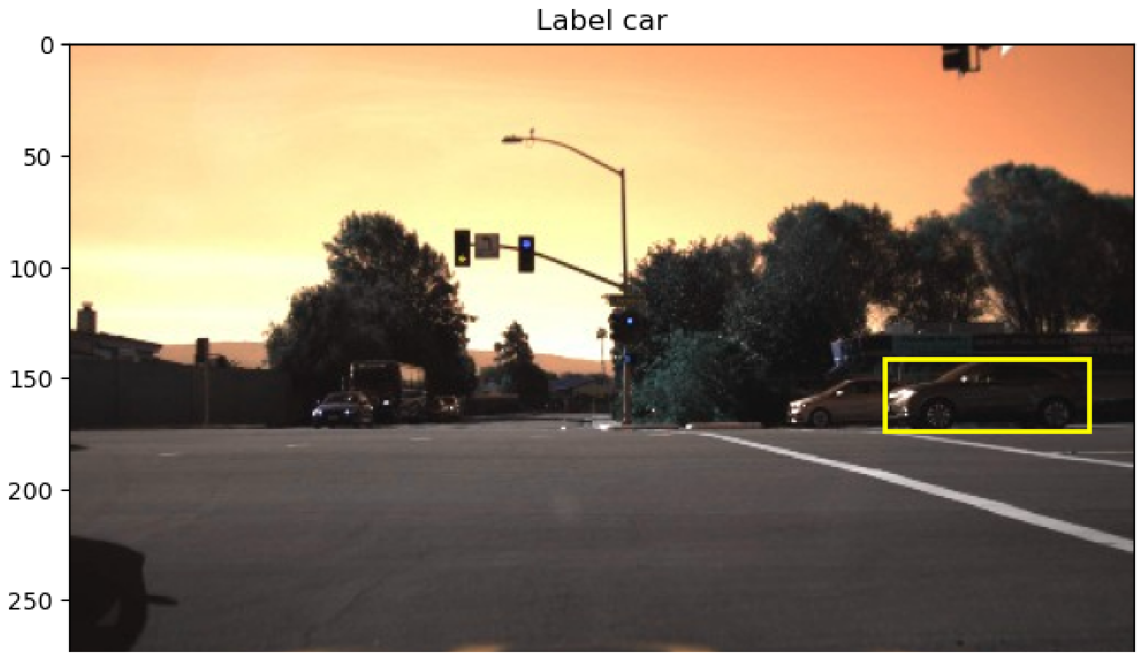
In [129]:

```python
for i in classes:

    xmin, xmax, ymin, ymax = boxes[i][0], boxes[i][1], boxes[i][2], boxes[

    plt.figure(figsize=(8, 10))
    plt.title("Label " + labels[i])
    plt.imshow(images[i])
    plt.gca().add_patch(plt.Rectangle((xmin, ymin), xmax-xmin, ymax-ymin,

    plt.show()
```



Model

In [130]:

```
!pip install ultralytics
```

```
Requirement already satisfied: ultralytics in /opt/conda/lib/python3.10/site-
packages (8.0.147)
Requirement already satisfied: matplotlib>=3.2.2 in /opt/conda/lib/python3.1
0/site-packages (from ultralytics) (3.7.1)
Requirement already satisfied: numpy>=1.22.2 in /opt/conda/lib/python3.10/sit
e-packages (from ultralytics) (1.23.5)
Requirement already satisfied: opencv-python>=4.6.0 in /opt/conda/lib/python
3.10/site-packages (from ultralytics) (4.8.0.74)
Requirement already satisfied: pillow>=7.1.2 in /opt/conda/lib/python3.10/sit
e-packages (from ultralytics) (9.5.0)
Requirement already satisfied: pyyaml>=5.3.1 in /opt/conda/lib/python3.10/sit
e-packages (from ultralytics) (6.0)
Requirement already satisfied: requests>=2.23.0 in /opt/conda/lib/python3.10/
site-packages (from ultralytics) (2.31.0)
Requirement already satisfied: scipy>=1.4.1 in /opt/conda/lib/python3.10/site
-packages (from ultralytics) (1.11.1)
Requirement already satisfied: torch>=1.7.0 in /opt/conda/lib/python3.10/site
-packages (from ultralytics) (2.0.0+cpu)
Requirement already satisfied: torchvision>=0.8.1 in /opt/conda/lib/python3.1
0/site-packages (from ultralytics) (0.15.1+cpu)
Requirement already satisfied: tqdm>=4.64.0 in /opt/conda/lib/python3.10/site
-packages (from ultralytics) (4.65.0)
Requirement already satisfied: pandas>=1.1.4 in /opt/conda/lib/python3.10/sit
e-packages (from ultralytics) (1.5.3)
Requirement already satisfied: seaborn>=0.11.0 in /opt/conda/lib/python3.10/s
ite-packages (from ultralytics) (0.12.2)
Requirement already satisfied: psutil in /opt/conda/lib/python3.10/site-packa
ges (from ultralytics) (5.9.3)
Requirement already satisfied: py-cpuinfo in /opt/conda/lib/python3.10/site-p
ackages (from ultralytics) (9.0.0)
Requirement already satisfied: contourpy>=1.0.1 in /opt/conda/lib/python3.10/
site-packages (from matplotlib>=3.2.2->ultralytics) (1.1.0)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.10/site
-packages (from matplotlib>=3.2.2->ultralytics) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /opt/conda/lib/python3.1
0/site-packages (from matplotlib>=3.2.2->ultralytics) (4.40.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /opt/conda/lib/python3.1
0/site-packages (from matplotlib>=3.2.2->ultralytics) (1.4.4)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.10/s
ite-packages (from matplotlib>=3.2.2->ultralytics) (21.3)
Requirement already satisfied: pyparsing>=2.3.1 in /opt/conda/lib/python3.10/
site-packages (from matplotlib>=3.2.2->ultralytics) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python
3.10/site-packages (from matplotlib>=3.2.2->ultralytics) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site
-packages (from pandas>=1.1.4->ultralytics) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/conda/lib/pyt
hon3.10/site-packages (from requests>=2.23.0->ultralytics) (3.1.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.10/site
-packages (from requests>=2.23.0->ultralytics) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/conda/lib/python3.1
0/site-packages (from requests>=2.23.0->ultralytics) (1.26.15)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.1
0/site-packages (from requests>=2.23.0->ultralytics) (2023.5.7)
Requirement already satisfied: filelock in /opt/conda/lib/python3.10/site-pac
kages (from torch>=1.7.0->ultralytics) (3.12.2)
Requirement already satisfied: typing-extensions in /opt/conda/lib/python3.1
```

```
0/site-packages (from torch>=1.7.0->ultralytics) (4.6.3)
Requirement already satisfied: sympy in /opt/conda/lib/python3.10/site-packag
es (from torch>=1.7.0->ultralytics) (1.12)
Requirement already satisfied: networkx in /opt/conda/lib/python3.10/site-pac
kages (from torch>=1.7.0->ultralytics) (3.1)
Requirement already satisfied: jinja2 in /opt/conda/lib/python3.10/site-packa
ges (from torch>=1.7.0->ultralytics) (3.1.2)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-pac
kages (from python-dateutil>=2.7->matplotlib>=3.2.2->ultralytics) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in /opt/conda/lib/python3.10/s
ite-packages (from jinja2->torch>=1.7.0->ultralytics) (2.1.3)
Requirement already satisfied: mpmath>=0.19 in /opt/conda/lib/python3.10/site
-packages (from sympy->torch>=1.7.0->ultralytics) (1.3.0)
```

In [131]:

```python
from ultralytics import YOLO
import PIL
from PIL import Image
from IPython.display import display
import os
import pathlib
```

In [132]:

```python
model = YOLO("yolov8m.pt")
```

In [133]:

```python
results=model.predict(source="/kaggle/input/self-driving-cars/images/14780
              save=True, conf=0.2,iou=0.5)
```

```
image 1/1 /kaggle/input/self-driving-cars/images/1478019956680248165.jpg: 416
x640 1 car, 1 truck, 518.0ms
Speed: 2.3ms preprocess, 518.0ms inference, 1.3ms postprocess per image at sh
ape (1, 3, 416, 640)
Results saved to runs/detect/predict6
```

In [134]:

```python
result = results[0]
box = result.boxes[0]
```

In [135]:

```python
for result in results:
    boxes = result.boxes  # Boxes object for bbox outputs
    masks = result.masks  # Masks object for segmentation masks outputs
    probs = result.probs  # Class probabilities for classification outputs
```

In [136]:

```python
cords = box.xyxy[0].tolist()
class_id = box.cls[0].item()
conf = box.conf[0].item()
print("Object type:", class_id)
print("Coordinates:", cords)
print("Probability:", conf)
```

```
Object type: 2.0
Coordinates: [124.69143676757812, 142.68826293945312, 148.7809295654297, 162.
10047912597656]
Probability: 0.7161698937416077
```

In [137]:

```
for box in result.boxes:
    class_id = result.names[box.cls[0].item()]
    cords = box.xyxy[0].tolist()
    cords = [round(x) for x in cords]
    conf = round(box.conf[0].item(), 2)
    print("Object type:", class_id)
    print("Coordinates:", cords)
    print("Probability:", conf)
    print("---")
```

```
Object type: car
Coordinates: [125, 143, 149, 162]
Probability: 0.72
---
Object type: truck
Coordinates: [141, 139, 157, 154]
Probability: 0.31
---
```

In [138]:

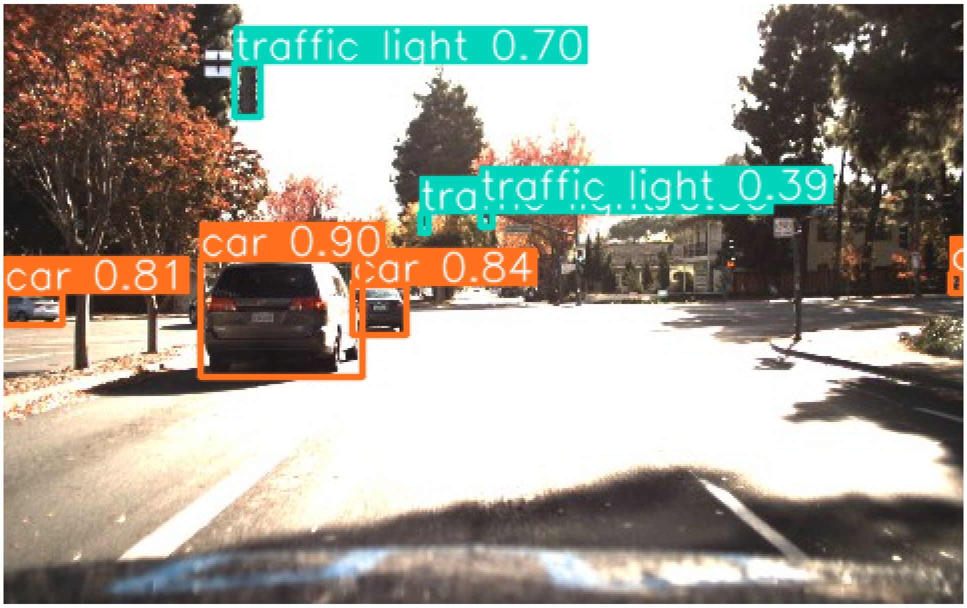
```
results1 = model.predict(source="/kaggle/input/self-driving-cars/images/14
                 save=True, conf=0.2,iou=0.5)

Results = results1[0]
```

```
image 1/1 /kaggle/input/self-driving-cars/images/1478020211690815798.jpg: 416
x640 4 cars, 3 traffic lights, 508.2ms
Speed: 2.2ms preprocess, 508.2ms inference, 1.1ms postprocess per image at sh
ape (1, 3, 416, 640)
Results saved to runs/detect/predict6
```

In [139]:

```
# Plotting results
plot = results1[0].plot()
plot = cv2.cvtColor(plot, cv2.COLOR_BGR2RGB)
display(Image.fromarray(plot))
```



In [ ]: