OBJECT DETECTION

S#	Name	Roll #/Section	SUBJECT
1.	HAMZA	FA/2020/BSCS/488 – (2)	COMPUTER VISION

Teacher Name: **MISS GHULSHAN SALEEM**

Department of Computer Science
Lahore Garrison University
Lahore

1. Introduction

- Definition: YOLO is an acronym for "You Only Look Once," a real-time object detection algorithm.

- Purpose: Efficiently detect and locate objects in images or video frames.

2. YOLO v8 Overview

- YOLO v8: The eighth version of the YOLO series, known for improved accuracy and speed.

- Architecture: Brief explanation of the neural network architecture used in YOLO v8.

3. Key Features

- One-Stage Detection: YOLO operates in a single pass through the neural network, making it faster than two-stage detectors.

-Anchor Boxes: YOLO uses anchor boxes to predict bounding boxes for different object sizes.

4. Object Detection Process

1. Input Image: Show an example input image.

2. Grid Division: Explain how the image is divided into a grid.

3. Bounding Box Prediction: YOLO predicts bounding boxes and confidence scores for each grid cell.

4. Non-Maximum Suppression: Describe the process of removing redundant bounding boxes.

5. YOLO v8 Advantages

- Real-time Performance: YOLO v8 is designed for fast and efficient object detection.
- High Accuracy: Discuss improvements in accuracy compared to previous YOLO versions.

6. Use Cases

- Autonomous Vehicles: YOLO v8 can be applied for real-time object detection in the context of self-driving cars.
- Surveillance Systems: Discuss how YOLO v8 is useful for monitoring and security applications.

7. Challenges and Limitations

- Small Object Detection: Address challenges related to detecting small objects.
- Complex Scenes: Discuss difficulties in crowded or complex scenes.

8. Future Developments

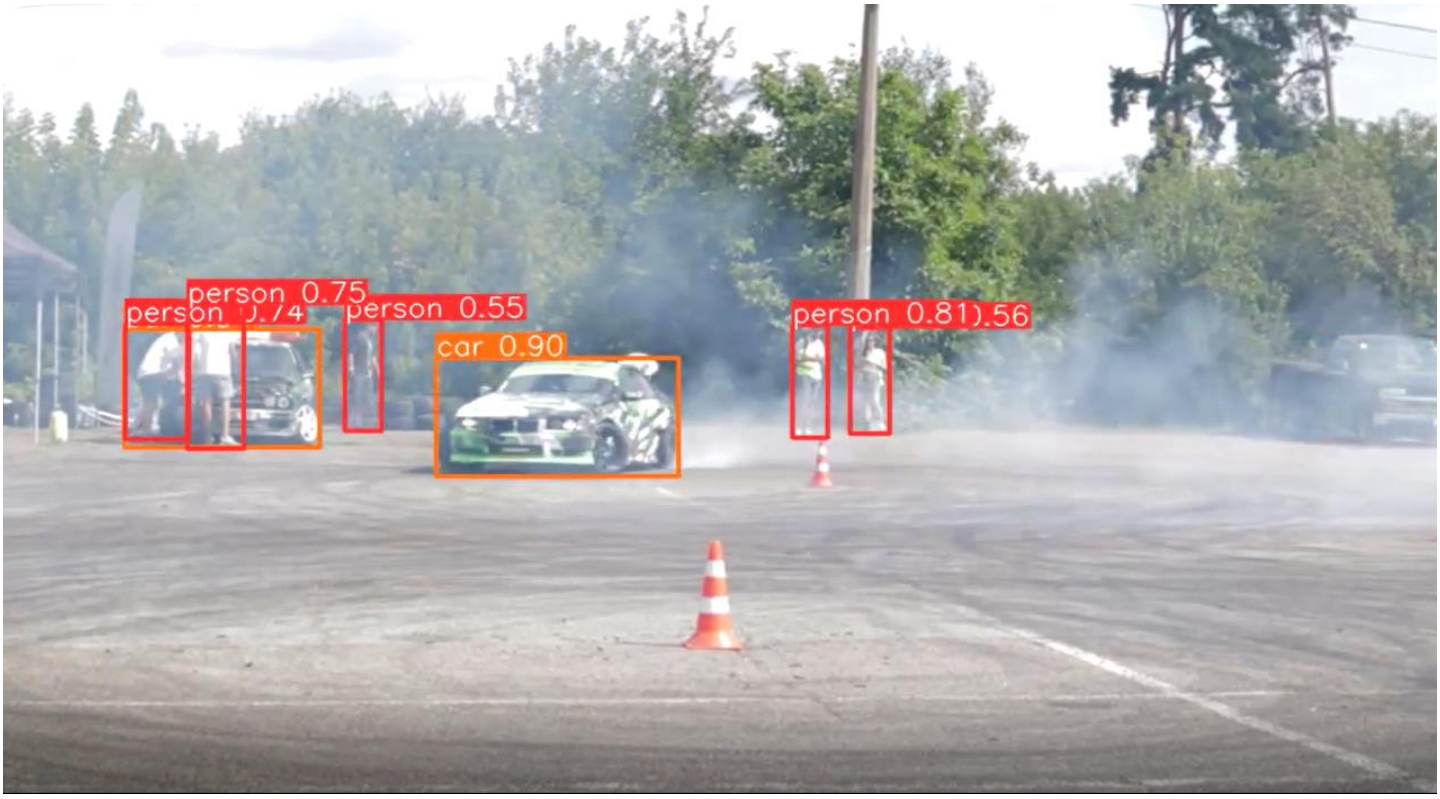
- Ongoing Research: Mention any ongoing research or future developments related to YOLO and object detection.

9. Conclusion

- Summarize the key points discussed in the presentation.
- Highlight YOLO v8's significance in the field of computer vision.

BFORE DETECTION

AFTER DETECTION



10. Screenshots of Google Colab

object Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect Colab AI

```
!pip install ultralytics -q
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 660.5/660.5 kB 6.7 MB/s eta 0:00:00
```

[ ] Start coding or generate with AI.

```
!yolo detect predict model=yolov8m.pt source="/content/drive/MyDrive/WhatsApp.mp4"
```

```
Downloading https://github.com/ultralytics/assets/releases/download/v0.0.0/yolov8m.pt to 'yolov8m.pt'...
100% 49.7M/49.7M [00:00<00:00, 167MB/s]
Ultralytics YOLOv8.0.227 🚀 Python-3.10.12 torch-2.1.0+cu118 CPU (Intel Xeon 2.20GHz)
YOLOv8m summary (fused): 218 layers, 25886080 parameters, 0 gradients, 78.9 GFLOPs

video 1/1 (1/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1 truck, 1161.7ms
video 1/1 (2/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 893.7ms
video 1/1 (3/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 906.2ms
video 1/1 (4/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 923.5ms
video 1/1 (5/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 880.5ms
video 1/1 (6/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 1404.0ms
video 1/1 (7/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 1454.7ms
video 1/1 (8/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 1430.1ms
video 1/1 (9/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 1432.8ms
video 1/1 (10/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 1031.1ms
video 1/1 (11/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 893.2ms
video 1/1 (12/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 916.0ms
video 1/1 (13/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 872.6ms
video 1/1 (14/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 876.1ms
video 1/1 (15/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 894.9ms
video 1/1 (16/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 904.1ms
video 1/1 (17/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 902.0ms
video 1/1 (18/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1 truck, 897.0ms
video 1/1 (19/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 1 car, 1 truck, 919.6ms
video 1/1 (20/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1 truck, 903.2ms
video 1/1 (21/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1 truck, 1345.9ms
video 1/1 (22/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1 truck, 1362.6ms
video 1/1 (23/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1414.0ms
video 1/1 (24/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1410.5ms
video 1/1 (25/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1145.7ms
video 1/1 (26/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 913.6ms
video 1/1 (27/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 910.1ms
video 1/1 (28/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 909.4ms
video 1/1 (29/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 919.2ms
video 1/1 (30/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 920.4ms
video 1/1 (31/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 1 car, 885.3ms
video 1/1 (32/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 888.7ms
```

object Detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect Colab AI

```
!ffmpeg -i {"/content/runs/detect/predict/WhatsApp.avi"} -vcodec libx264 {"final.mp4"}
```

```
ffmpeg version 4.4.2-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg developers
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.22.04.1 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu --arch=amd64 --enable-gpl --disable-stripping --enable-gnutls --enable-ladspa --enable-liba
  libavutil      56. 70.100 / 56. 70.100
  libavcodec     58.134.100 / 58.134.100
  libavformat    58. 76.100 / 58. 76.100
  libavdevice    58. 13.100 / 58. 13.100
  libavfilter     7.110.100 /  7.110.100
  libswscale      5.  9.100 /  5.  9.100
  libswresample   3.  9.100 /  3.  9.100
  libpostproc    55.  9.100 / 55.  9.100
Input #0, avi, from '/content/runs/detect/predict/WhatsApp.avi':
  Metadata:
    software        : Lavf59.27.100
  Duration: 00:00:19.24, start: 0.000000, bitrate: 15374 kb/s
  Stream #0:0: Video: mjpeg (Baseline) (MJPG / 0x47504A4D), yuvj420p(pc, bt470bg/unknown/unknown), 1024x576, 15392 kb/s, 29 fps, 29 tbr, 29 tbn, 29 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (mjpeg (native) -> h264 (libx264))
Press [q] to stop, [?] for help
[libx264 @ 0x55fdfa30d340] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2 AVX FMA3 BMI2 AVX2
[libx264 @ 0x55fdfa30d340] profile High, level 3.1, 4:2:0, 8-bit
[libx264 @ 0x55fdfa30d340] 264 - core 163 r3060 5db6aa6 - H.264/MPEG-4 AVC codec - Copyleft 2003-2021 - http://www.videolan.org/x264.html - options: cabac=1 ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00 mixed_ref=1 me_ra
Output #0, mp4, to 'final.mp4':
  Metadata:
    software        : Lavf59.27.100
    encoder         : Lavf58.76.100
  Stream #0:0: Video: h264 (avc1 / 0x31637661), yuvj420p(pc, bt470bg/unknown/unknown, progressive), 1024x576, q=2-31, 29 fps, 14848 tbn
    Metadata:
      encoder         : Lavc58.134.100 libx264
    Side data:
      cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: N/A
frame=  558 fps= 24 q=-1.0 Lsize=    3147kB time=00:00:19.13 bitrate=1347.3kbits/s speed=0.811x
video:3140kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.233059%
[libx264 @ 0x55fdfa30d340] frame I:3     Avg QP:20.23  size: 60762
[libx264 @ 0x55fdfa30d340] frame P:153   Avg QP:22.73  size: 13974
[libx264 @ 0x55fdfa30d340] frame B:402   Avg QP:26.62  size:  2225
[libx264 @ 0x55fdfa30d340] consecutive B-frames:  3.0%  1.8%  2.7% 92.5%
[libx264 @ 0x55fdfa30d340] mb I  I16..4:  2.1% 91.9%  5.9%
[libx264 @ 0x55fdfa30d340] mb P  I16..4:  1.8% 15.2%  1.0%  P16..4: 35.2% 14.3%  9.8%  0.0%  0.0%    skip:22.7%
[libx264 @ 0x55fdfa30d340] mb B  I16..4:  0.3%  0.8%  0.2%  B16..8: 31.9%  2.4%  0.7%  direct: 0.6%  skip:63.1%  L0:41.1% L1:55.5% BI: 3.4%
[libx264 @ 0x55fdfa30d340] 8x8 transform intra:81.6% inter:74.8%
[libx264 @ 0x55fdfa30d340] coded y,uvDC,uvAC intra: 73.7% 62.8% 11.5% inter: 11.1% 9.2% 2.8%
[libx264 @ 0x55fdfa30d340] i16 v,h,dc,p: 24% 53%  5% 18%
[libx264 @ 0x55fdfa30d340] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 14% 44% 27%  2%  2%  2%  4%  2%  4%
[libx264 @ 0x55fdfa30d340] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 32% 35% 11%  3%  4%  3%  5%  4%  4%
[libx264 @ 0x55fdfa30d340] i8c dc,h,v,p: 36% 40% 21%  3%
```

```
video 1/1 (24/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1410.5ms
video 1/1 (25/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 1145.7ms
video 1/1 (26/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 913.6ms
video 1/1 (27/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 6 persons, 2 cars, 910.1ms
video 1/1 (28/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 909.4ms
video 1/1 (29/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 919.2ms
video 1/1 (30/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 920.4ms
video 1/1 (31/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 1 car, 885.3ms
video 1/1 (32/558) /content/drive/MyDrive/WhatsApp.mp4: 384x640 7 persons, 2 cars, 888.7ms
```

11. Explain the lines of codes

This line of code appears to be an FFmpeg command used for video processing. Let's break it down:

- !ffmpeg: This is the command to execute FFmpeg, a powerful multimedia processing tool.

- -i {"/content/runs/detect/predict/WhatsApp.avi"}: This specifies the input file. In this case, it's a video file located at the specified path ("/content/runs/detect/predict/WhatsApp.avi").

- -vcodec libx264: This sets the video codec to libx264, which is a popular and efficient video compression codec.

- {"final.mp4"}: This part specifies the output file. In this case, the processed video will be saved as "final.mp4" in the current working directory.

So, the entire command is instructing FFmpeg to take the input video file ("WhatsApp.avi"), encode it using the libx264 codec, and save the result as an output video file named "final.mp4."

Note that the use of curly braces ({}) around the input file path may be specific to certain programming environments or scripts.

hese lines of code seem to be related to installing the Ultralytics library and then using YOLO (You Only Look Once) for object detection on a video. Let's break it down:

1. !pip install ultralytics -q: This line installs the Ultralytics library using the Python package manager, pip. The -q flag makes the installation process quiet, suppressing unnecessary output.

2. !yolo detect predict model=yolov8m.pt source="/content/drive/MyDrive/WhatsApp.mp4": This line uses the YOLO command-line interface (CLI) to perform object detection on a video. Breaking it down:

- !yolo: Invokes the YOLO command-line tool.

- detect predict: Specifies the YOLO command to detect and predict objects in the given input.

- model=yolov8m.pt: Specifies the YOLO model to be used for detection. In this case, it's yolov8m.pt.

- source="/content/drive/MyDrive/WhatsApp.mp4": Specifies the source of the input, which is a video file located at "/content/drive/MyDrive/WhatsApp.mp4".

This code is likely used in a Python environment, and the exclamation mark (!) before the commands indicates that these are shell commands executed in a Jupyter Notebook or a similar interactive environment. The Ultralytics library is used to interact with and deploy YOLO models conveniently.