July 2022

UARK NACME-Google AMLI Summer Bootcamp

Capstone Project

Applied Machine Learning Intensive
Student Detection
Tracking & Counting

UNIVERSITY OF ARKANSAS

NACME

NATIONAL ACTION COUNCIL FOR MINORITIES IN ENGINEERING

VT

Google

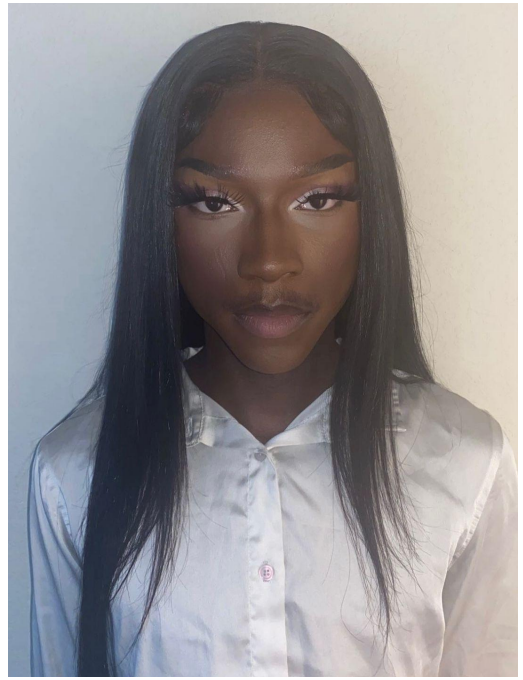
EYG introductions

Ellion Dison
Senior in Computer Engineering

UNIVERSITY OF ARKANSAS

Yasser Hassan
Senior in Electrical Engineering

VIRGINIA TECH



Gabriel Young
Junior in Computer Science

UNIVERSITY OF ARKANSAS

overall goal

build a real-time human detection system that accurately tracks and counts the people in our classroom

the demo

overview

human detection, tracking, and counting

detection is the task of locating every instance of a human or object present in an image. this is done by searching all frames at every scale and comparing them with the known patterns of the people or objects.

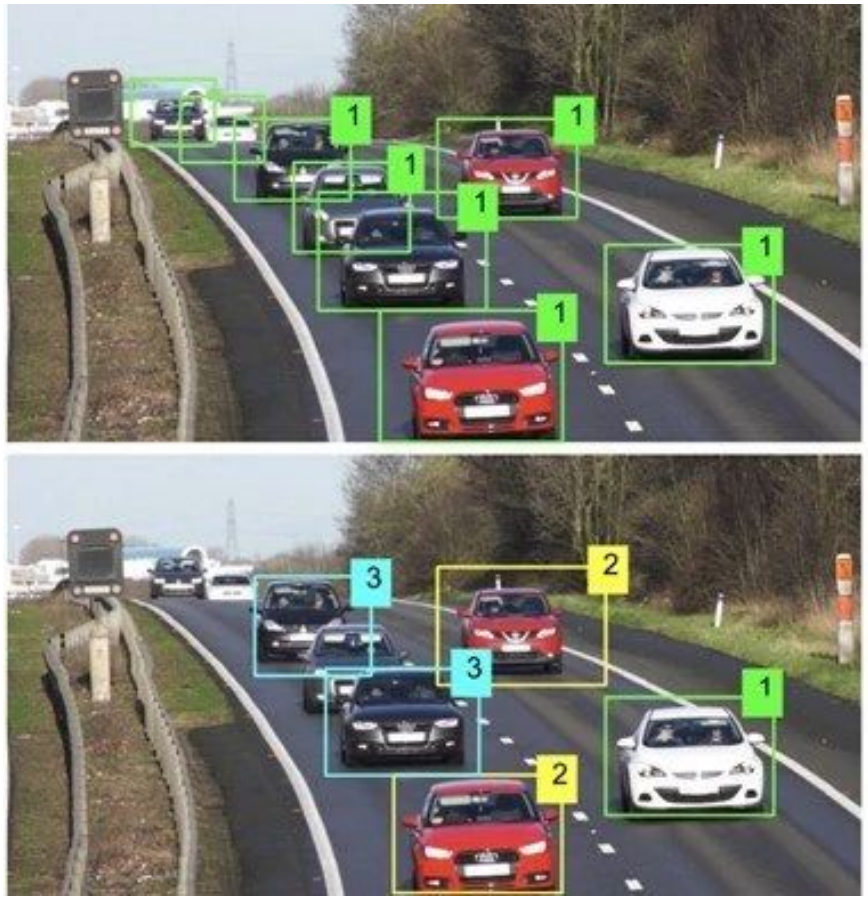
tracking works by using a deep learning program that takes an initial set of object detections and develops a unique identification for each of the initial detections and then tracks the detected objects as they move around frames in a video.

object tracking algorithms

People often get object detection and tracking confused. Object detection is simply about locating and classifying all known objects in a frame. Object tracking is about locking onto a particular moving object(s) in real-time and giving it a unique ID.

There are several object tracking algorithms like ROLO, Deepsort, Byte Track, and MDNet.

Deepsort is a MOT algorithm and it is one of the of the most widely used.



Typical Object Detection Algorithm

Vs.

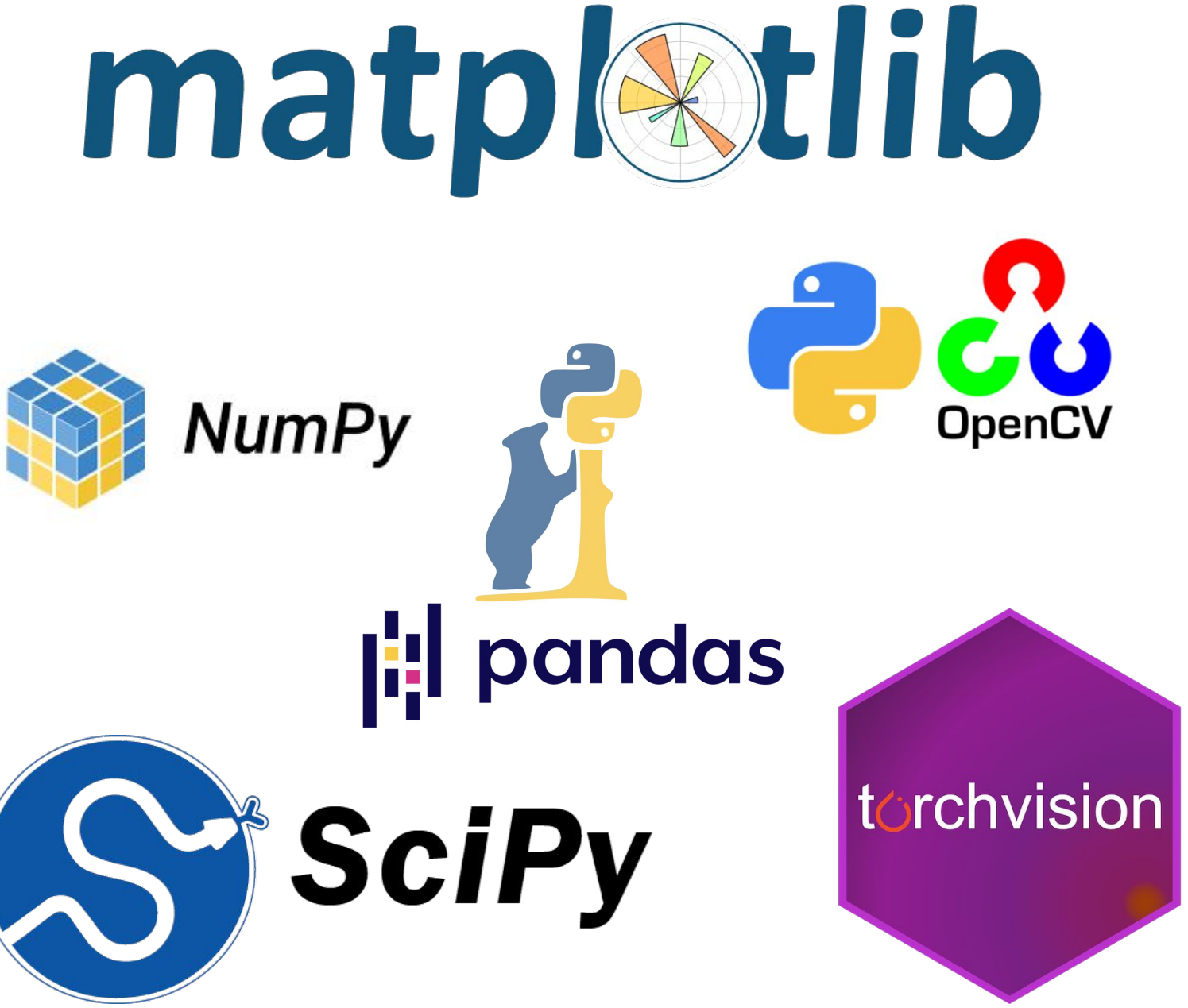
Typical Object Tracking Algorithm

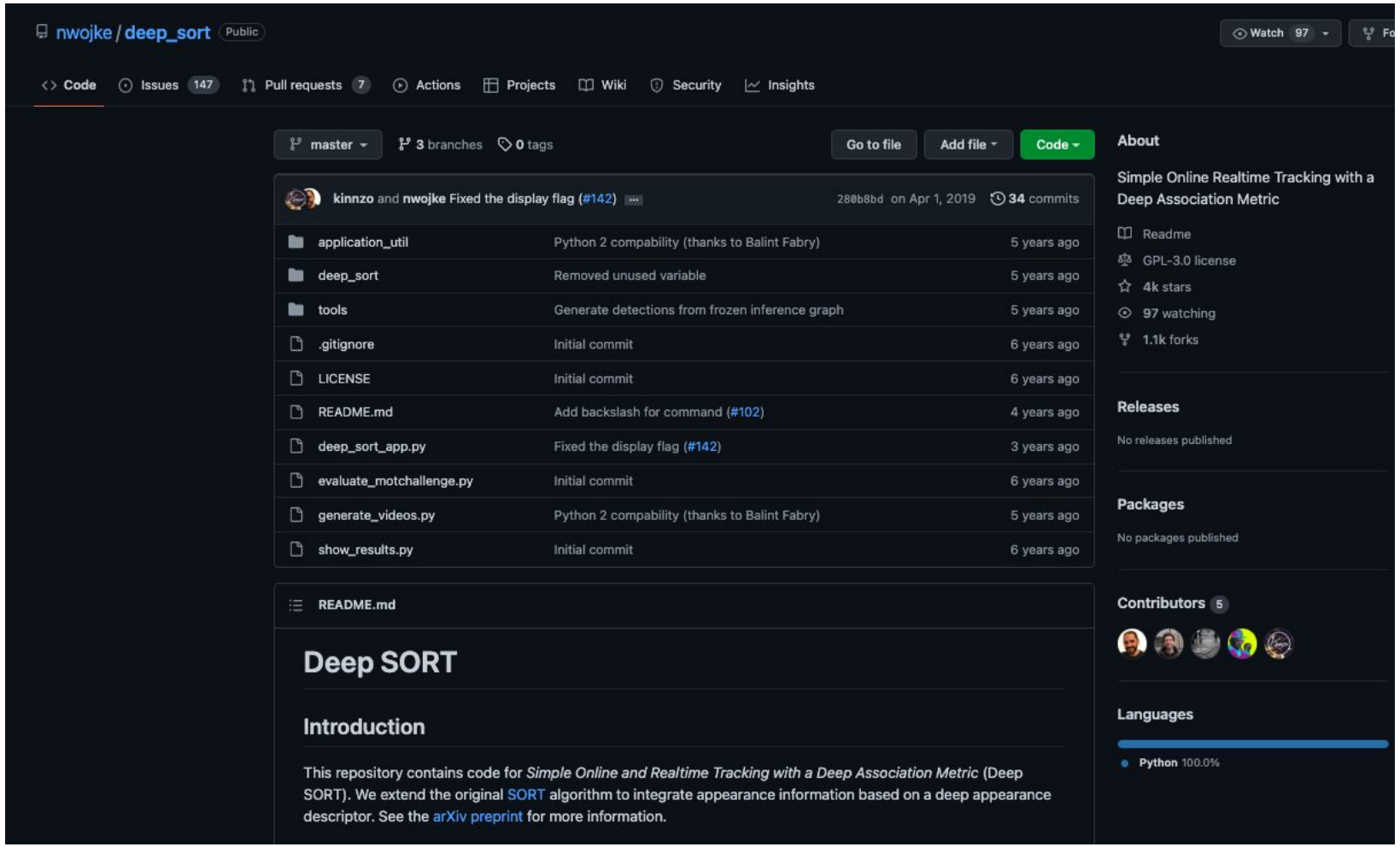
Results (output)

tools used

libraries

- *matplotlib*
- *numpy*
- *opencv-python*
- *pandas*
- *pillow*
- *pyYAML*
- *requests*
- *seaborn*
- *scipy*
- *torch*
- *torchvision*
- *tqdm*





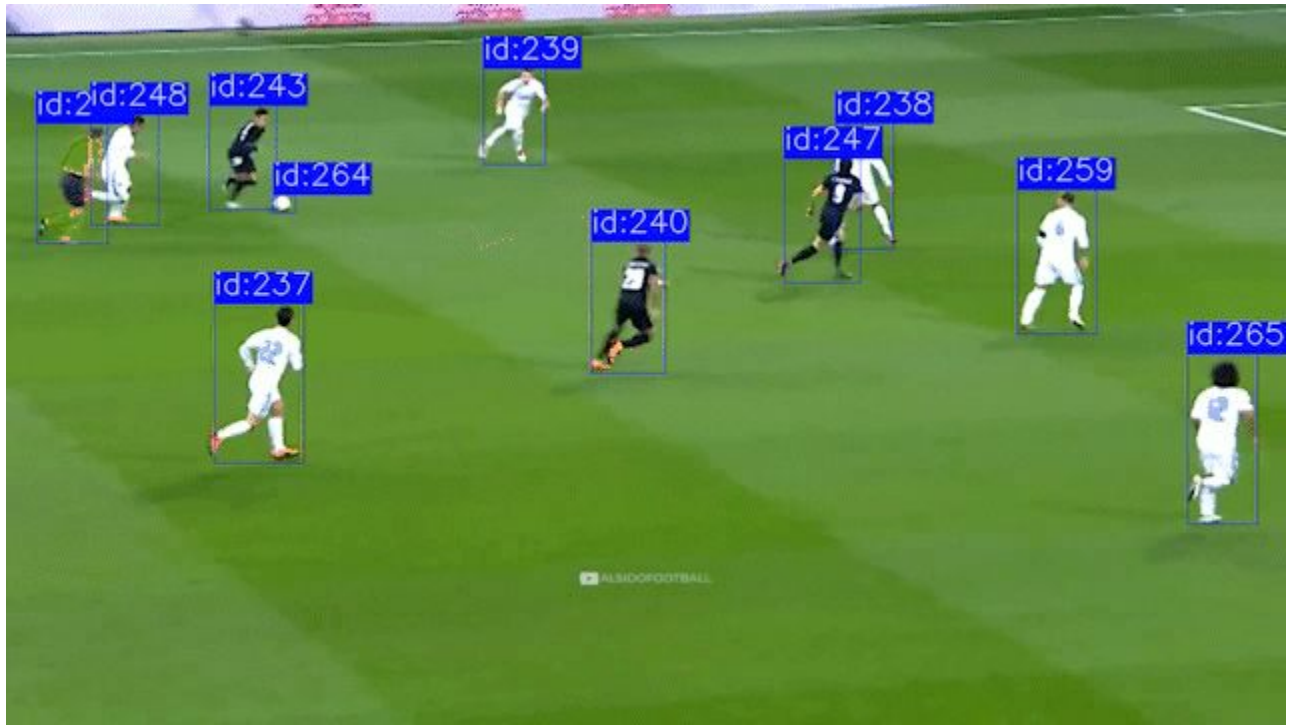
code repository

we used nwojke's deep sort repository as a basis for our project's code

this repository contained a base model that we reformatted

quick background

Deep sort extends the SORT (simple online and real-time tracking with a deep association metric) algorithm by implementing deep learning. SORT is highly effective with tracking precision and accuracy but causes a high volume of ID switches. ID Switching can occur when an object disappear and reappear in frames or when objects have similar features. When this happens the initial unique ID is destroyed and a new one is created. This is where deep sort comes in by combining motion and appearance descriptors to a lot for a better association metric thus reducing the number of ID switches.



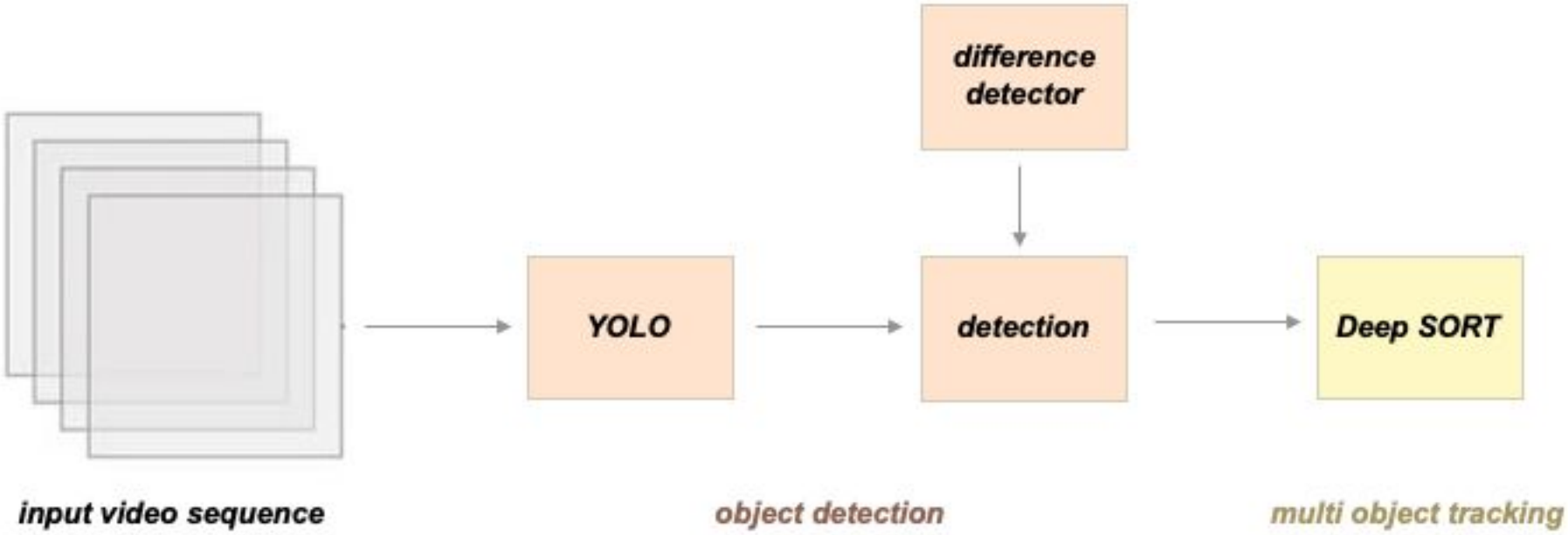


Diagram Explained

The deep sort algorithm utilizes a separate algorithm known as yolo which uses CNN (convolutional neural networks) to perform detection. The yolo algorithm feeds the detections into deep sort in order to create an accurate real-time tracker.

code examples

```
47  # initialize deep sort
48  model_filename = 'model_data/mars-small128.pb'
49  encoder = gdet.create_box_encoder(model_filename, batch_size=1)
50  # calculate cosine distance metric
51  metric = nn_matching.NearestNeighborDistanceMetric("cosine", max_cosine_distance, nn_budget)
52  # initialize tracker
53  tracker = Tracker(metric)
54
```

```
# draw bbox on screen
    color = colors[int(track.track_id) % len(colors)]
    color = [i * 255 for i in color]
    cv2.rectangle(frame, (int(bbox[0]), int(bbox[1])), (int(bbox[2]), int(bbox[3])), color, 2)
    cv2.rectangle(frame, (int(bbox[0]), int(bbox[1]-30)), (int(bbox[0])+(len(class_name)+len(str(track.track_id)))*17, int(bbox[1])), colo
    cv2.putText(frame, class_name + "-" + str(track.track_id),(int(bbox[0]), int(bbox[1]-10)),0, 0.75, (255,255,255),2)
```

```
        # Print results
        for c in det[:, -1].unique():
            n = (det[:, -1] == c).sum()  # detections per class
            cv2.putText(im0, "Objects being tracked: {}".format(n), (5, 35),
                        cv2.FONT_HERSHEY_COMPLEX_SMALL, 1.5, (46, 26, 71), 2)
            s += f"{n} {names[int(c)]}{'s' * (n > 1)}, "  # add to string
```

acquiring the data

challenge

- ***our model took several hours to finish running***
- ***model detects computer and counts it as a person***

solution

- ***trim the video, lower the resolution, and resize the video***
- ***display the confidence score of all parties then make overall score higher than the computer's specific score to stop its detection***

next: training the data to our model

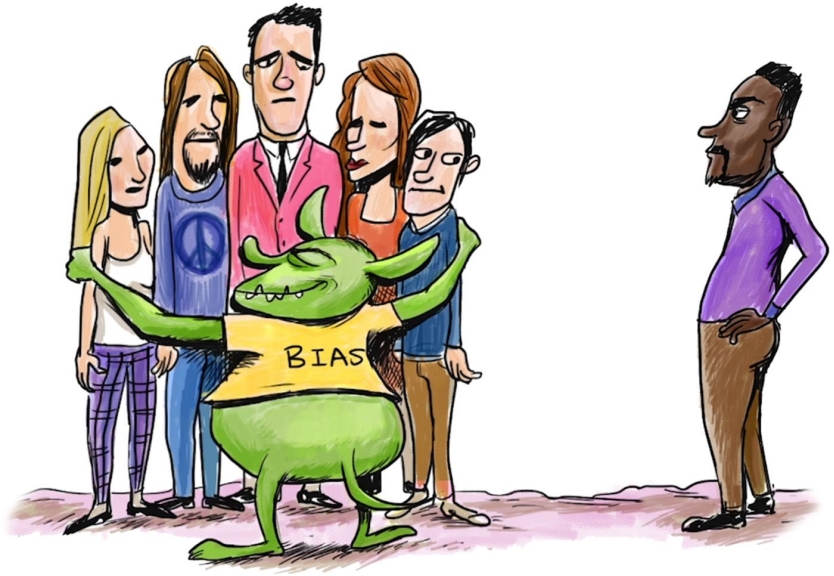
model adjustments with yolo

yolo v4

yolo v5

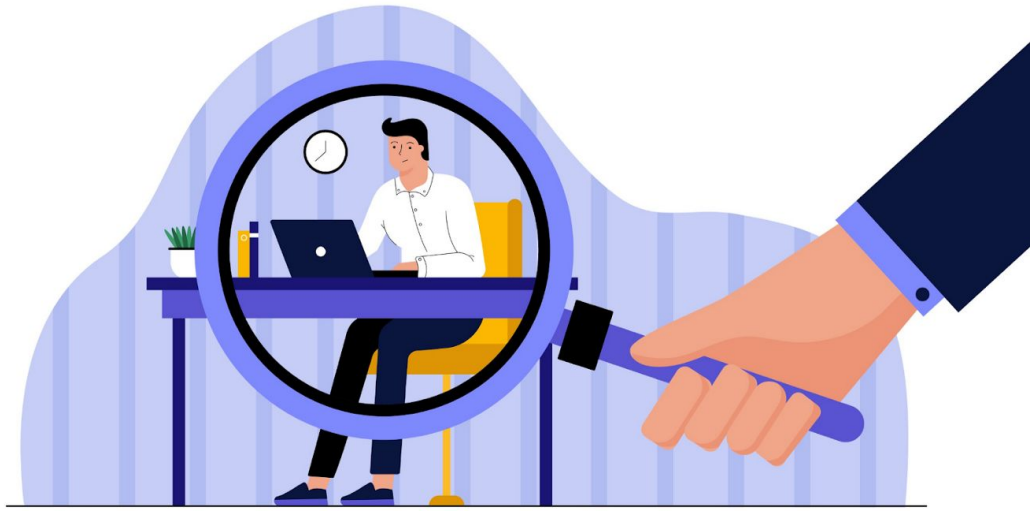
motivations

- *use for detection in large classroom settings such as the one's on our respective campuses to help professors monitor exams.*
- *standard cctv usage*
- *tracking in nursing homes*



ethical bias

models like ours can struggle with detecting individuals with a darker complexion which presents the problem of racial bias



ethical consideration

with algorithms like this, the concept of safety and privacy should always be considered for consumers

Conclusions and Findings

- ***Our deepsort tracker was able to accurately track and count the number of students in the classroom using yolov4 and yolov5***
- ***yolov5 was about 3x faster than yolov4 at processing videos/live feed***
- ***A confidence threshold of .5 was good for filtering out possible false detections***
- ***Overall, yolov5 had less switching and it was a little more accurate than yolov4 at detecting and tracking humans***

live webcam demo

challenge: live feed from webcam using yolov4 was very slow and laggy

solution: implementing yolo v5 to increase performance speed

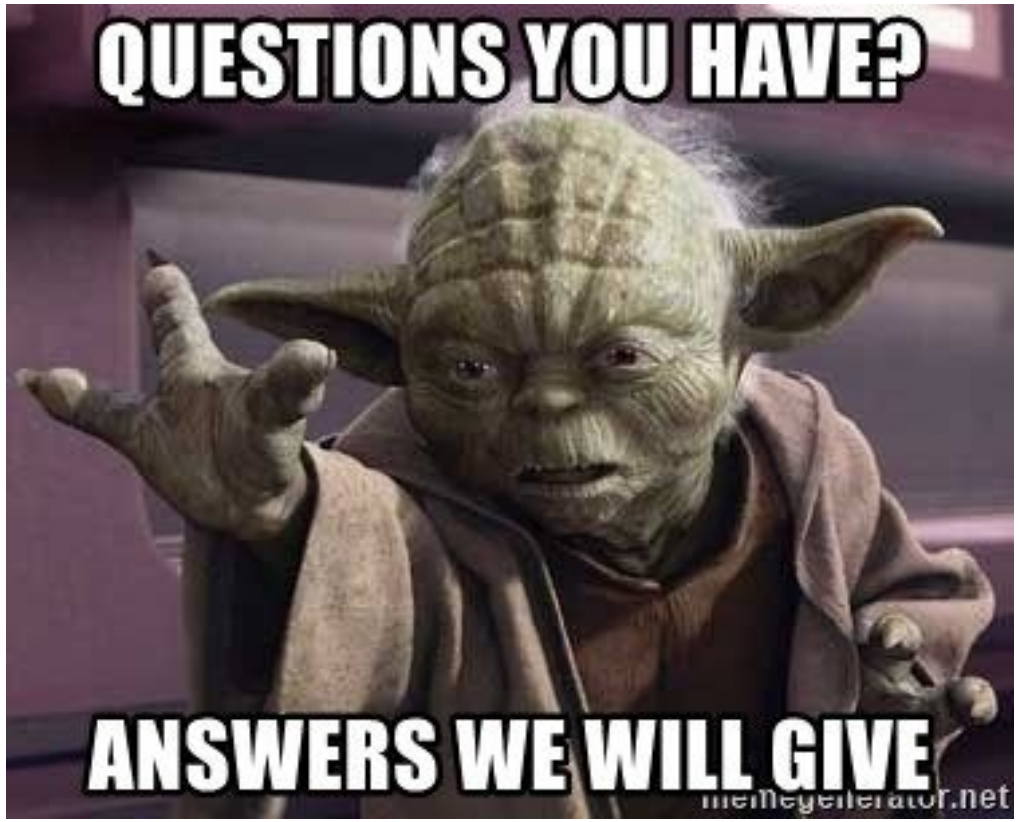
Acknowledgments

For this summer experience, we would like to thank:

NACME | National Action Council for Minorities in Engineering
The University of Arkansas
Google

and we'd also like to thank these specific individuals:

Michele Lezama
Dr. Thi Hoang Ngan Le
Hyekang Joo | Kevin
Dr. Bryan Hill
Teresa Simpson


QUESTIONS YOU HAVE?
ANSWERS WE WILL GIVE
memegenerator.net