SINGLE OBJECT TRACKING

Group13

LIST OF CONTENTS

Single Object Tracking (SOT) involves locating and monitoring a specific target across consecutive video frames. Techniques encompass traditional methods like filters and modern deep learning approaches. Challenges include handling occlusions, scale changes, and diverse lighting conditions. Evaluation metrics include precision, recall, and Intersection over Union (IoU). SOT advancements integrate deep learning for more accurate and robust tracking, applicable in surveillance and robotics. SOT continues to evolve for enhanced real-time tracking in various applications.

LITERATURE REVIEW

YOLOv8 in tracking elucidates its novel algorithm that integrates the YOLO object detection method with tracking techniques, enhancing accuracy and resilience in dynamic scenarios.

01 “A Research on CSR-DCF Tracking Algorithm based on YOLO Detection”

Proposes an algorithm merging YOLO with CSR-DCF for target tracking, enhancing accuracy through self-attention integration, closed-loop control, and improved feature representation, showcasing superiority over traditional tracking methods.

02 "Research on Video Tracking Algorithm Based on Yolo Target Detection"

Introduces a multi-target tracking algorithm combining YOLOv4 and SORT, emphasizing precise detection and efficient tracking, showcasing improved accuracy in dynamic scenarios through advanced architectural elements and experiments on the PETS09-S2L1 dataset.

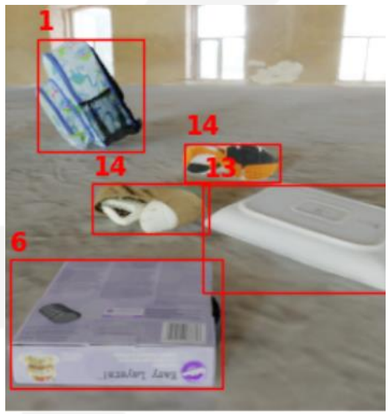
03 “Object Tracking by detection using YOLO and SORT”

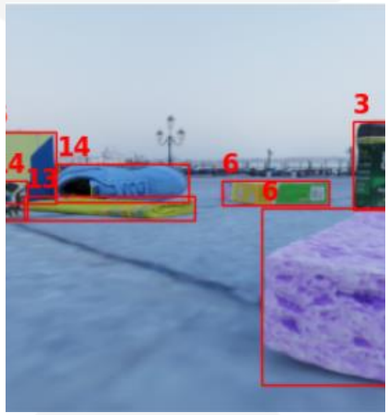
Explores object tracking through YOLO and SORT, detailing a tracking-by-detection approach, custom dataset training, and experimental results focusing on precision and recall metrics for six distinct object classes.

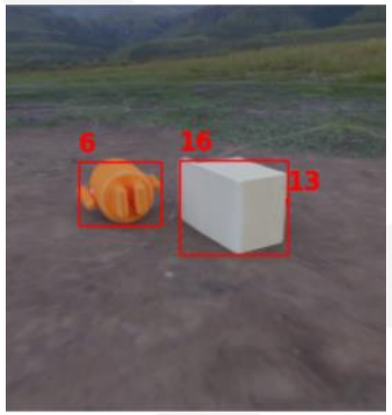
DATA PREPROCCSEING

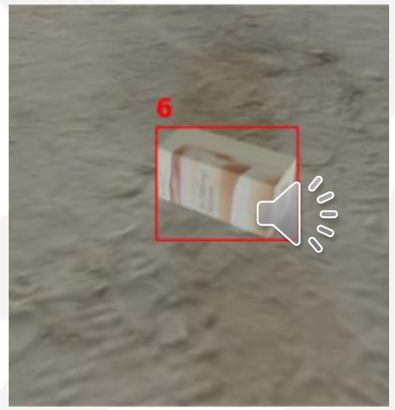
	image	target	image_path	VideoID	ImageID	Class
0	[[[238, 200, 175], [235, 199, 171], [234, 199,...	[[74, 103, 129, 178, 13], [67, 80, 108, 130, 6...	fixed_random_rotate_video_0_image_0	0	0	fixed_random_rotate
1	[[[238, 200, 175], [235, 199, 171], [234, 199,...	[[76, 104, 128, 178, 13], [67, 84, 89, 132, 6]...	fixed_random_rotate_video_0_image_1	0	1	fixed_random_rotate
2	[[[238, 200, 175], [235, 199, 171], [234, 199,...	[[77, 104, 129, 179, 13], [68, 87, 89, 132, 6]...	fixed_random_rotate_video_0_image_2	0	2	fixed_random_rotate
3	[[[238, 200, 175], [235, 199, 171], [234, 199,...	[[79, 104, 129, 180, 13], [68, 89, 90, 132, 6]...	fixed_random_rotate_video_0_image_3	0	3	fixed_random_rotate
4	[[[238, 200, 175], [235, 199, 171], [234, 199,...	[[83, 103, 129, 182, 13], [69, 89, 94, 132, 6]...	fixed_random_rotate_video_0_image_4	0	4	fixed_random_rotate
...	...	...	...	...	...	...
2155	[[[180, 75, 46], [174, 73, 47], [165, 68, 43],...	[[88, 87, 147, 146, 6], [85, 68, 147, 143, 16]...	rotation_rotate_video_29_image_19	29	19	rotation_rotate
2156	[[[194, 87, 54], [193, 86, 54], [193, 84, 53],...	[[88, 94, 146, 142, 6], [87, 70, 144, 138, 16]...	rotation_rotate_video_29_image_20	29	20	rotation_rotate
2157	[[[194, 89, 51], [194, 89, 51], [194, 89, 51],...	[[89, 102, 144, 137, 6], [85, 74, 144, 138, 16...	rotation_rotate_video_29_image_21	29	21	rotation_rotate
2158	[[[188, 84, 49], [188, 85, 49], [191, 87, 51],...	[[88, 103, 144, 143, 6], [82, 69, 144, 139, 16...	rotation_rotate_video_29_image_22	29	22	rotation_rotate
2159	[[[193, 88, 53], [191, 87, 51], [192, 88, 52],...	[[86, 99, 144, 150, 6], [80, 65, 143, 123, 16]...	rotation_rotate_video_29_image_23	29	23	rotation_rotate

2160 rows × 6 columns





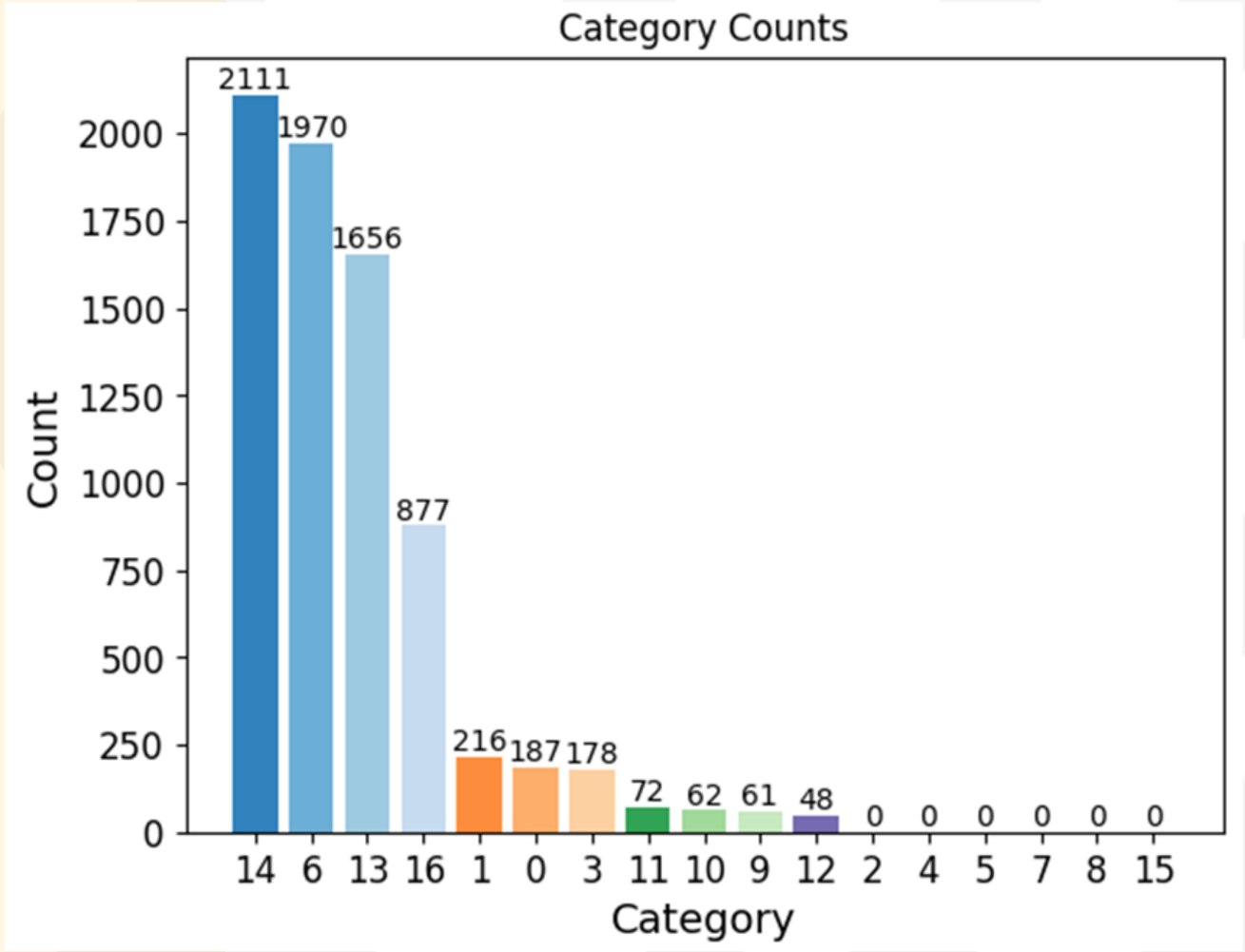




OBJECTS COUNTS

Each object possesses a unique ID among the 16 identified objects. Each frame may contain multiple instances of the same object or entirely different objects within it.

```
objectNames = {
    0: 'Action Figures',
    1: 'Bag',
    2: 'Board Games',
    3: 'Bottles and Cans and Cups',
    4: 'Camera',
    5: 'Car Seat',
    6: 'Consumer Goods',
    7: 'Hat',
    8: 'Headphones',
    9: 'Keyboard',
    10: 'Legos',
    11: 'Media Cases',
    12: 'Mouse',
    13: 'None',
    14: 'Shoe',
    15: 'Stuffed Toys',
    16: 'Toys',
    17: 'Blocker'
}
```

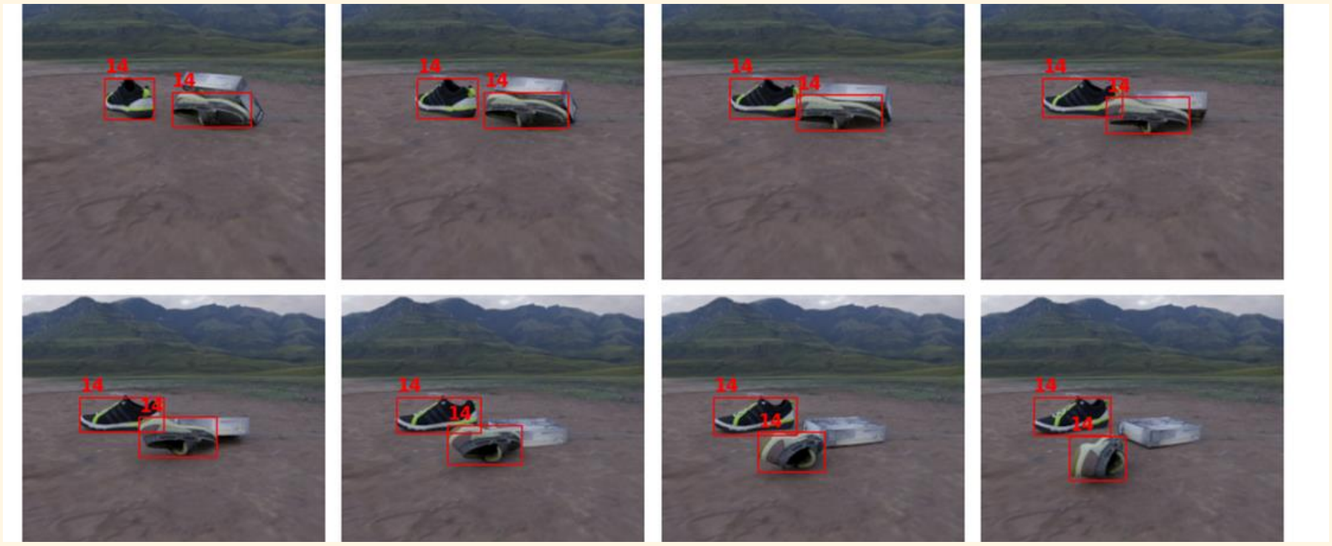


OBJECT SELECTION & NORMALIZATION

Object ID =14--> Shoe

```
new_df['normalize_target'] = new_df['target'].apply(lambda x: [normalize_bbox(box, image_shape) for box in x])
new_df
```

	image	target	image_path	VideoID	ImageID	Class	normalize_target
0	[[[236, 232, 230], [234, 231, 228], [229, 226,...	[[97, 126, 126, 193, 14], [85, 69, 120, 111, 14]]	linear_movement_rotate_video_4_image_0	4	0	linear_movement_rotate	[(0.623046875, 0.435546875, 0.26171875, 0.1132...
1	[[[236, 232, 229], [232, 228, 227], [228, 226,...	[[97, 120, 127, 191, 14], [85, 63, 119, 114, 14]]	linear_movement_rotate_video_4_image_1	4	1	linear_movement_rotate	[(0.607421875, 0.4375, 0.27734375, 0.1171875, ...
2	[[[234, 230, 228], [230, 227, 226], [228, 225,...	[[99, 114, 129, 186, 14], [85, 57, 119, 117, 14]]	linear_movement_rotate_video_4_image_2	4	2	linear_movement_rotate	[(0.5859375, 0.4453125, 0.28125, 0.1171875, 14...
3	[[[232, 228, 227], [230, 227, 226], [227, 225,...	[[101, 106, 132, 176, 14], [85, 52, 118, 119, ...	linear_movement_rotate_video_4_image_3	4	3	linear_movement_rotate	[(0.55078125, 0.455078125, 0.2734375, 0.121093...
4	[[[231, 227, 227], [230, 227, 227], [226, 224,...	[[103, 98, 137, 164, 14], [86, 48, 116, 119, 14]]	linear_movement_rotate_video_4_image_4	4	4	linear_movement_rotate	[(0.51171875, 0.46875, 0.2578125, 0.1328125, 1...
...	...	...	...	...	...	...	...
1089	[[[205, 198, 177], [199, 193, 171], [195, 190,...	[[99, 96, 185, 158, 14], [71, 167, 90, 212, 14]]	fixed_random_rotate_video_27_image_22	27	22	fixed_random_rotate	[(0.49609375, 0.5546875, 0.2421875, 0.3359375,...
1090	[[[205, 198, 177], [199, 193, 171], [195, 190,...	[[101, 101, 187, 156, 14], [70, 168, 89, 210, ...	fixed_random_rotate_video_27_image_23	27	23	fixed_random_rotate	[(0.501953125, 0.5625, 0.21484375, 0.3359375, ...
1091	[[[84, 81, 83], [85, 82, 84], [87, 84, 85], [9...	[[95, 155, 99, 156, 14]]	fixed_random_rotate_video_28_image_2	28	2	fixed_random_rotate	[(0.607421875, 0.37890625, 0.00390625, 0.01562...
1092	[[[84, 81, 83], [85, 82, 84], [87, 84, 85], [9...	[[84, 146, 106, 152, 14]]	fixed_random_rotate_video_28_image_3	28	3	fixed_random_rotate	[(0.58203125, 0.37109375, 0.0234375, 0.0859375...
1093	[[[84, 81, 83], [85, 82, 84], [87, 84, 85], [9	[[77, 133, 110, 145, 14]]	fixed_random_rotate_video_28_image_4	28	4	fixed_random_rotate	[(0.54296875, 0.365234375, 0.046875, 0.1289062



DATA SPLITTING

	Class	VideoID	Count	Count_train	Count_valid
0	fixed_random_rotate	0	24	16	8
1	fixed_random_rotate	3	24	16	8
2	fixed_random_rotate	6	24	16	8
3	fixed_random_rotate	7	24	16	8
4	fixed_random_rotate	9	18	12	6
5	fixed_random_rotate	10	24	16	8
6	fixed_random_rotate	15	24	16	8
7	fixed_random_rotate	17	24	16	8
8	fixed_random_rotate	18	24	16	8
9	fixed_random_rotate	21	24	16	8
10	fixed_random_rotate	22	24	16	8
11	fixed_random_rotate	25	24	16	8
12	fixed_random_rotate	26	24	16	8
13	fixed_random_rotate	27	24	16	8
14	fixed_random_rotate	28	3	2	1
15	linear_movement_rotate	4	24	16	8
16	linear_movement_rotate	5	24	16	8
17	linear_movement_rotate	7	24	16	8
18	linear_movement_rotate	9	24	16	8
19	linear_movement_rotate	10	24	16	8
20	linear_movement_rotate	13	24	16	8
21	linear_movement_rotate	15	24	16	8
22	linear_movement_rotate	18	24	16	8
23	linear_movement_rotate	20	24	16	8
24	linear_movement_rotate	23	24	16	8
25	linear_movement_rotate	24	24	16	8
26	linear_movement_rotate	25	24	16	8

70%

Training/Video

30%

Validation/Video

730 FRAMES

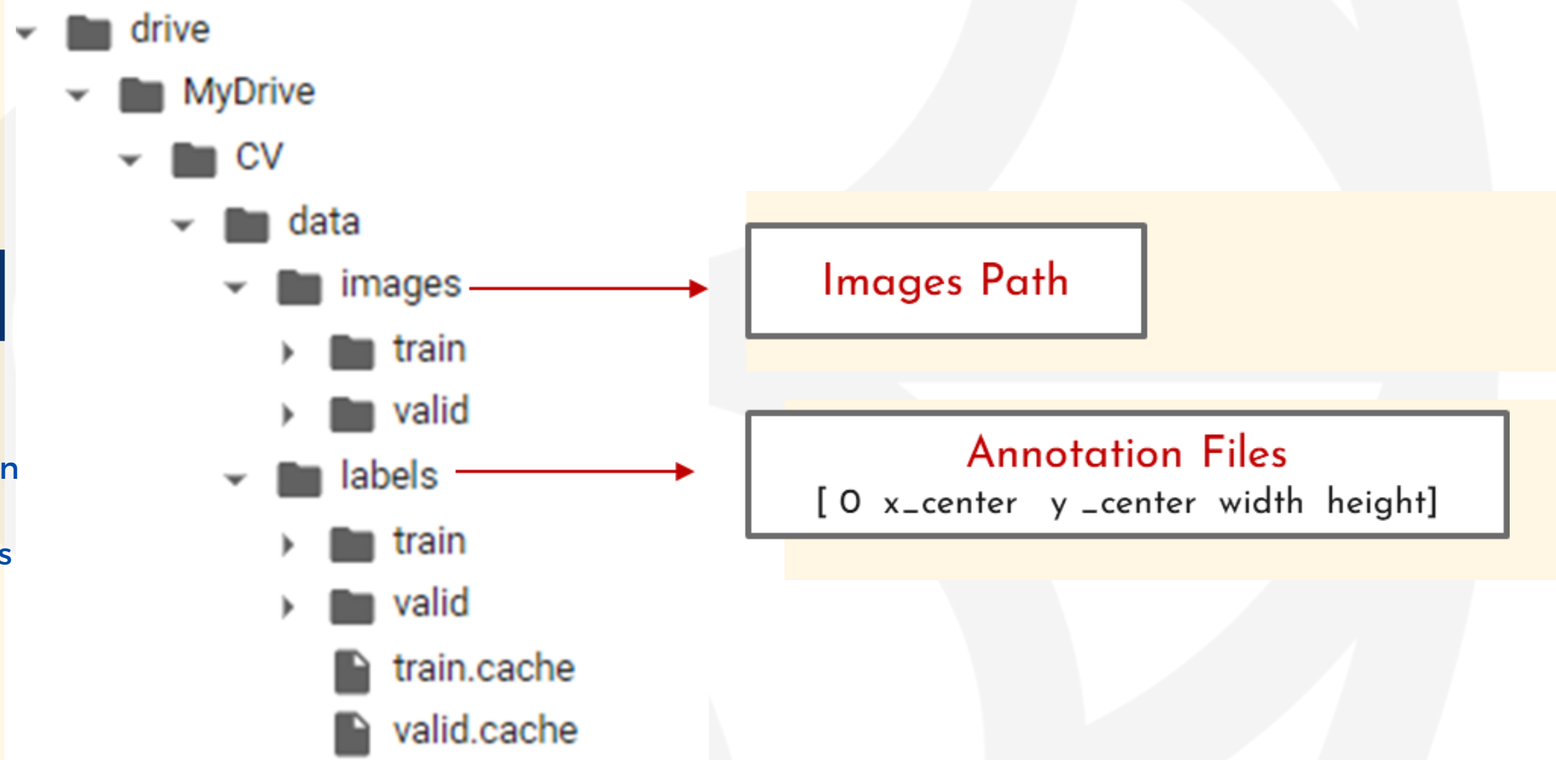
Total Frames of Training

364 FRAMES

Total Frames of Testing

DATA PREPARATION

For YOLOv8, an image annotation file for each frame is required in a .txt format.
The file consists of 'n' rows, corresponding to the count of objects detected in that frame. Each row includes data in the format '0 x_center y_center width height' for each object, where '0' indicates single-object tracking.



YOLOV8 MODEL

YOLOv8, a state-of-the-art object detection model, utilizes its detection capabilities as a means to assist in the tracking process. In tracking scenarios, YOLOv8 first detects and identifies objects within a frame, then uses the detected objects' information (such as their positions, classes, and features) to establish and maintain their identities across subsequent frames. This detection-based approach helps to initialize and sustain the tracking of objects, enabling the system to monitor and follow these objects across a video sequence or in a dynamic environment.

01 DETECTION

02 TRACKING

YOLOV8 VERSIONS

- N
- S
- M
- L
- FIXED MODEL

LEARNING RATE

- 0.01 LR
- 0.05 LR
- 0.001 LR

TRACKERS

- BOTSORT
- BYTE TRACKER

YOLOV8 VERSIONS

SUMMRAY

N	225 LAYERS, 3157200 PARAMETERS, 3157184 GRADIENTS, 8.9 GFLOPS
S	225 LAYERS, 11166560 PARAMETERS, 11166544 GRADIENTS, 28.8 GFLOPS
M	295 LAYERS, 25902640 PARAMETERS, 25902624 GRADIENTS, 79.3 GFLOPS
L	365 LAYERS, 43691520 PARAMETERS, 43691504 GRADIENTS, 165.7 GFLOPS
FIXED MODEL	ADDING LAYERS ON S

ENHANCING OBJECT DETECTION: ARCHITECTURAL MODIFICATIONS IN YOLOV8S FOR ABLATION STUDY

The YOLOv8 architecture employs various layers like convolutional layers, C2f, and SPPF for feature extraction. Modifications in the customized YOLOv8s model introduce two new layers after SPPF, enhancing pattern detection.

```yaml
backbone:
  # [from, repeats, module, args]
  - [-1, 1, Conv, [64, 3, 2]]  # 0-P1/2
  - [-1, 1, Conv, [128, 3, 2]]  # 1-P2/4
  - [-1, 3, C2f, [128, True]]
  - [-1, 1, Conv, [256, 3, 2]]  # 3-P3/8
  - [-1, 6, C2f, [256, True]]
  - [-1, 1, Conv, [512, 3, 2]]  # 5-P4/16
  - [-1, 6, C2f, [512, True]]
  - [-1, 1, Conv, [1024, 3, 2]]  # 7-P5/32
  - [-1, 3, C2f, [1024, True]]
  - [-1, 1, SPPF, [1024, 5]]  # 9
  - [-1, 1, Conv, [1024, 3, 1]]  # New Conv layer after SPPF
  - [-1, 1, C2f, [1024, True]]  # New C2f block
```

CONSTANCE PARMETERS FOR DETECTION

EPOCHS=40
BATCH SIZE =16
CONFIDENCE >0.7

```
config = {
    "path":"/content/drive/MyDrive/CV/data",
    "train": "images/train",
    "val":"images/valid",

    "nc": 1,  # Number of classes
    'names': {0: objectNames[objectTrackID]},
    "batch-size": 16,
    "subdivisions": 4,
    "width": 256,
    "height": 256,
    "channels": 3,
    "momentum": 0.949,
    "decay": 0.0005,
    "angle": 0,
    "saturation": 1.5,
    "exposure": 1.5,
    "hue": 0.1,
    "learning-rate": LearningRate,
    "burn-in": 1000,
    "max-batches": 500500,
    "policy": "steps",
    "steps": "400000,450000",
    "scales": "0.1,0.1",
    "letter-box": 0
}

# Save the configuration to a YAML file
with open("yolo_config.yaml", "w") as yaml_file:
    yaml.dump(config, yaml_file)

# Load YOLOv8n
model = YOLO(Yoloversion)  # load a pretrained YOLOv8n detection model
model.train(data='/content/yolo_config.yaml', epochs=40)  # train the model
```

YOLOv8 to detect using .Train

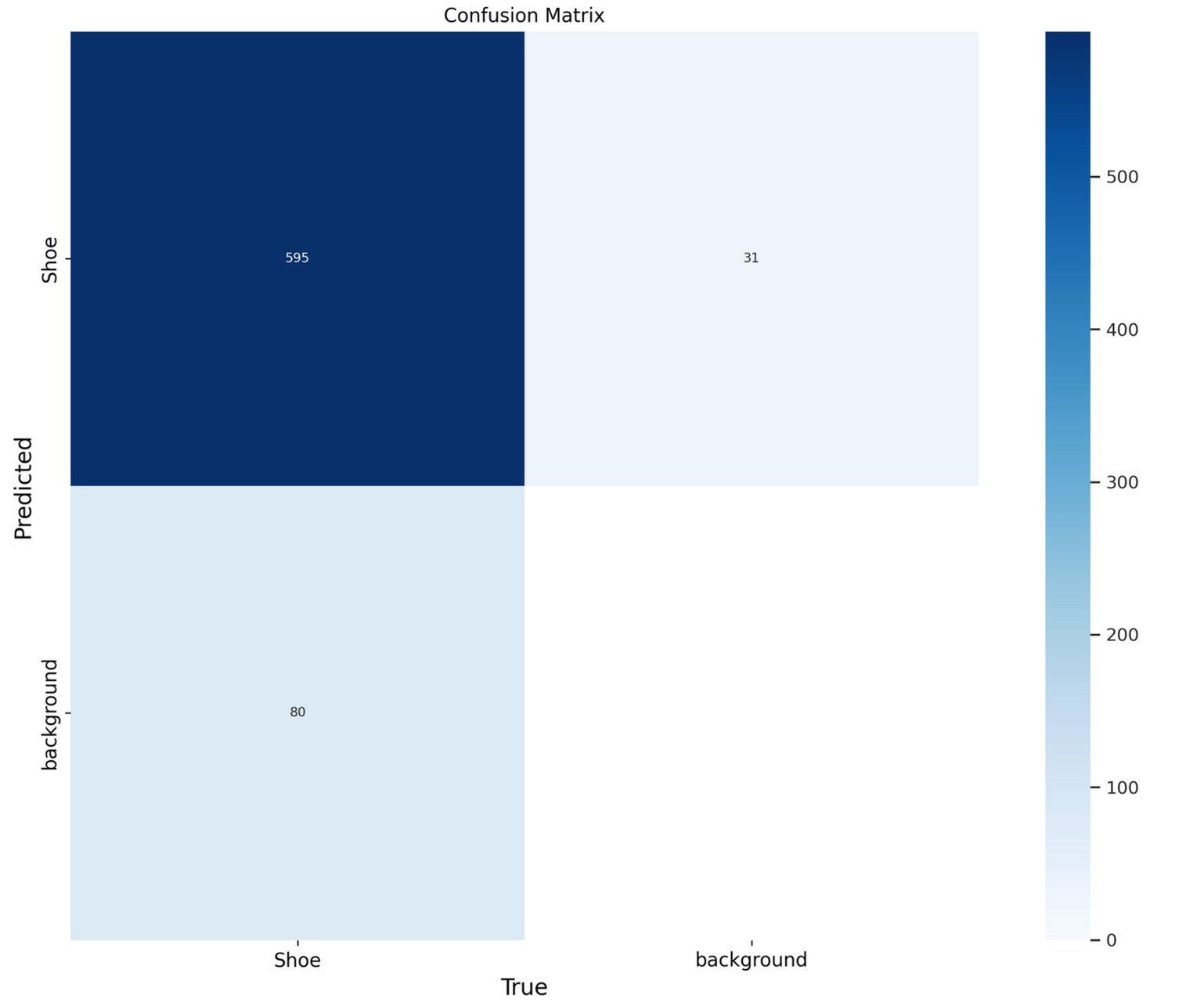
DETECTION SAMPLE

DETECTION EVALUATION USING MAP50-95

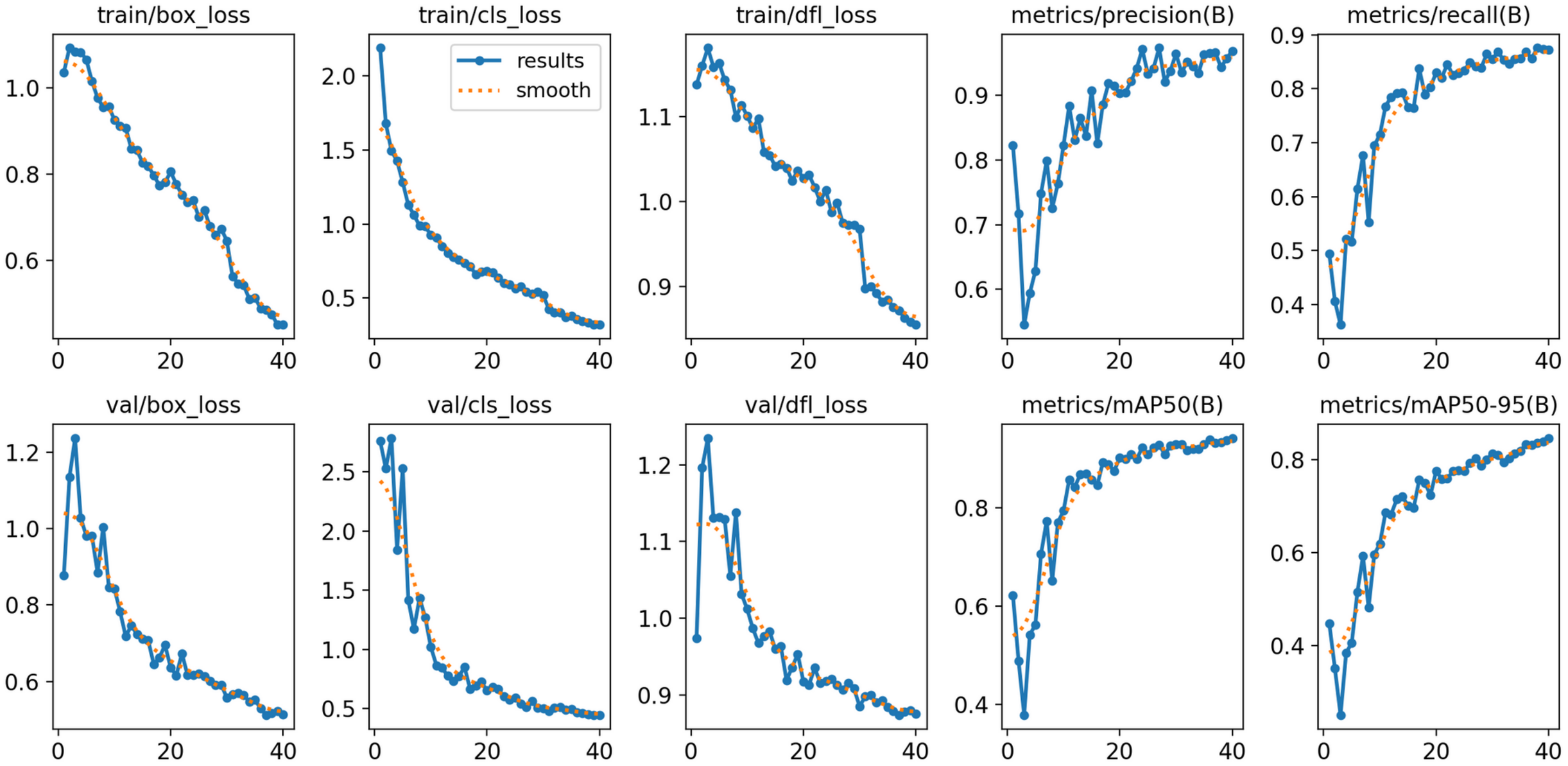
MODELS || LR	BOX PRECISON	BOX RECALL	mAP50	mAP50-95
N	0.05 LR= 96.8% 0.01 LR= 96.8% 0.001 LR = 96.8%	0.5 LR = 87.3% 0.1 LR = 87.3% 0.001 LR = 87.3%	0.05 LR= 94.2% 0.01 LR= 94.2% 0.001 LR = 94.2%	0.05 LR = 84.6% 0.01 LR = 84.6% 0.001 LR = 84.6%
S	0.05 LR= 97.1% 0.01 LR= 97.1% 0.001 LR = 97.1%	0.05 LR= 85% 0.01 LR= 85% 0.001 LR= 85%	0.05 LR = 94.1% 0.01 LR = 94.1% 0.001 LR = 94.1%	0.05 LR = 84.4% 0.01 LR = 84.4% 0.001 LR = 84.4%
M	0.05 LR = 94.9% 0.01 LR = 94.9% 0.001 LR = 94.9%	0.05 LR = 86.5% 0.01 LR = 86.5% 0.001 LR = 86.5%	0.05 LR = 93% 0.01 LR = 93% 0.001 LR = 93%	0.05 LR = 83.3% 0.01 LR = 83.3% 0.001 LR = 83.3%
L	0.05 LR = 92.4% 0.01 LR = 92.4% 0.001 LR = 92.4%	0.05 LR = 84.3% 0.01 LR = 84.3% 0.001 LR = 84.3%	0.05 LR = 91.4% 0.01 LR =91.4% 0.001 LR =91.4%	0.05 LR =81.6% 0.01 LR =81.6% 0.001 LR =81.6%
FIXED MODEL	0.05 LR = 83.6% 0.01 LR = 83.6% 0.001 LR = 83.6%	0.05 LR = 79.4% 0.01 LR = 79.4% 0.001 LR = 79.4%	0.05 LR = 86% 0.01 LR = 86% 0.001 LR = 86%	0.05 LR = 69.3% 0.01 LR = 69.3% 0.001 LR = 69.3%

CONFUSION MATRIX OF THE CHAMPION MODEL

YOLOV8N 0.001 LR



LOSS FUNCTION



TRACKING

```
frame = cv2.resize(frame, (256, 256))

results = model.track(frame, persist=True, tracker= Tracker)
boxes = results[0].boxes.xywh.cpu().numpy().astype(int)
```

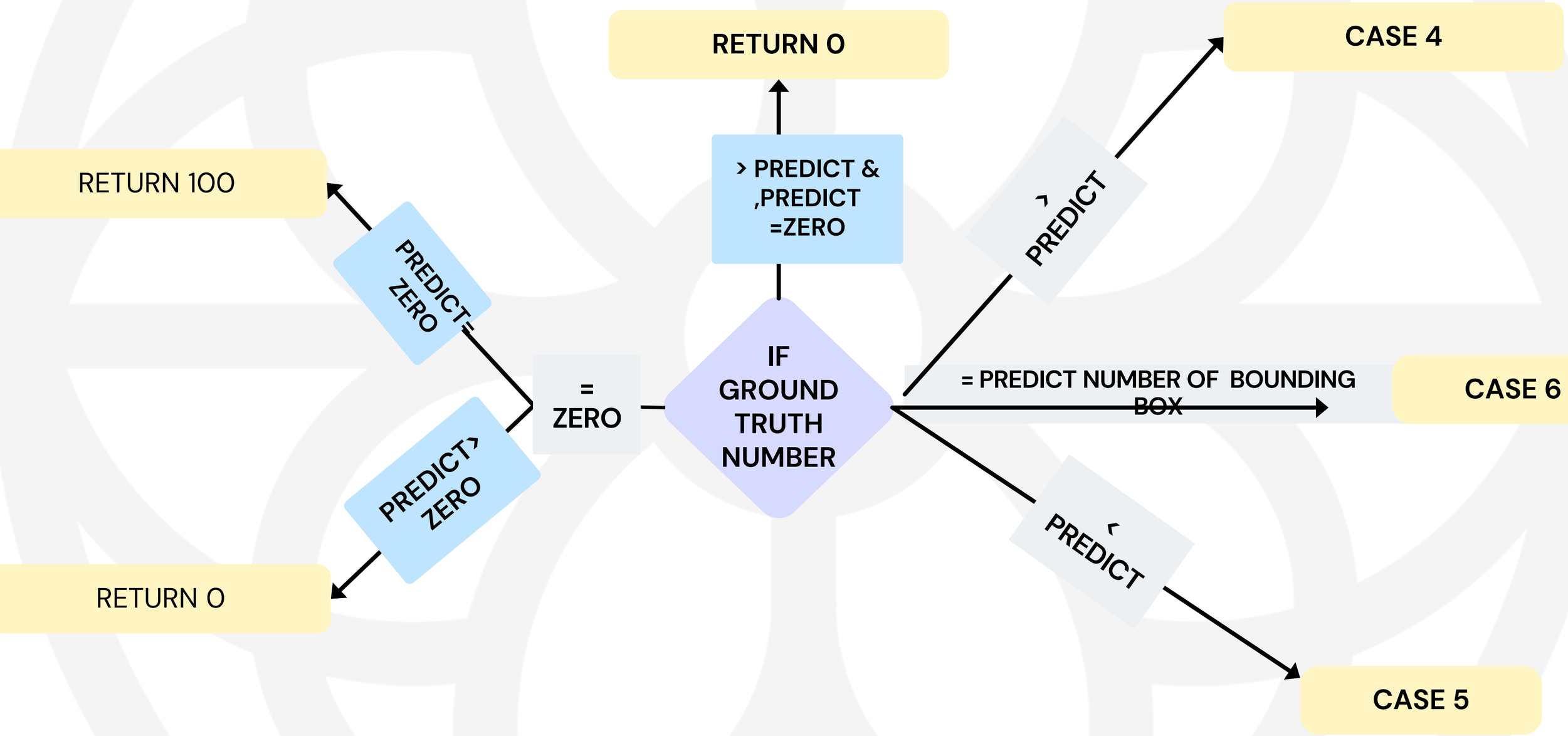
TRACKING EVALUATION

USING IOU

MODELS || TRACKERS	BOT SORT	BYTE TRACKER
X	0.05 LR= 81.7% 0.01 LR= 81.7% 0.001 LR = 81.7%	0.5 LR = 68.9% 0.1 LR = 68.9% 0.001 LR = 68.9%
S	0.05 LR= 81.6% 0.01 LR= 81.6% 0.001 LR= 81.6%	0.05 LR= 69.1% 0.01 LR= 69.1% 0.001 LR= 69.1%
M	0.05 LR = 79.2% 0.01 LR = 79.2% 0.001 LR = 79.2%	0.05 LR = 67.5% 0.01 LR = 67.5% 0.001 LR = 67.5%
L	0.05 LR = 78,57% 0.01 LR = 78,57% 0.001 LR = 78,57%	0.05 LR = 65.16% 0.01 LR = 65.16% 0.001 LR = 65.16%
FIXED MODEL	0.05 LR = 69.7% 0.01 LR = 69.7% 0.001 LR = 69.7%	0.05 LR = 58.8% 0.01 LR = 58.8% 0.001 LR = 58.8%

THE HANDLING OF EVALUATION CASES

BETWEEN THE GROUND TRUTH BOUNDING BOXES AND PREDICT FROM TRACKING



CASE 1	BOTH GROUND TRUTH AND PREDICTED BOXES ARE EMPTY: RETURNS A SCORE OF 100, SIGNIFYING A FRAME WITHOUT DETECTED OBJECTS AND DEVOID OF GROUND TRUTH ANNOTATIONS.
CASE 2	NO GROUND TRUTH, BUT PREDICTED BOXES EXIST: YIELDS A SCORE OF 0, INDICATING OBJECT DETECTION BY THE MODEL WITHOUT AVAILABLE GROUND TRUTH ANNOTATIONS FOR COMPARISON.
CASE 3	GROUND TRUTH EXISTS, BUT NO PREDICTED BOXES: RETURNS A SCORE OF 0, INDICATING THE MODEL'S FAILURE TO DETECT OBJECTS DESPITE GROUND TRUTH ANNOTATIONS.
CASE 4	GROUND TRUTH(GT) ARE MORE THAN THE PREDICTED BOXES: CALCULATES THE BEST IOU FOR EACH GT BOX AGAINST ALL PREDICTED BOXES AND GATHER THEM IN LIST AND GET THE SUMMATION THEN DIVIDE IT BY THE NUMBER OF GT BOXES.
NOTE	NOTE: : CALCULATES THE BEST IOU FOR EACH GT BOX BECAUSE THE INPUT MIGHT NOT BE SORTED, WHICH MEANS THAT THE PREDICTED OBJECTS ARE NOT SORTED AS SAME AS THE GROUND TRUTH OBJECTS. ALSO DIVIDE IT BY THE NUMBER OF GT BOXES HERE IS ESSENTIAL BECAUSE THE THIS WILL DECREASE THE IOU BECAUSE OF MISSING SOME OBJECTS.
CASE 5	·GROUND TRUTH(GT) ARE LESS THAN THE PREDICTED BOXES: CALCULATES THE BEST IOU FOR EACH PREDICTED BOX AGAINST ALL GT BOXES AND GATHER THEM IN LIST AND GET THE SUMMATION THEN DIVIDE IT BY THE NUMBER OF PREDICTED BOXES.
CASE 6	ALIGNED GROUND TRUTH AND PREDICTED BOXES WITH THE SAME COUNTS: COMPUTES IOU FOR EACH PREDICTED BOX AGAINST ALL GROUND TRUTH BOXES THEN GET THE AVERAGE.

Conclusion

Learning rate

Model complexity

Data importance

THANKS
For Watching

Group13