`

Harnessing Pre-trained ResNet for YOLO-Based Object Detection: A Loss Function Journey

Ren-Di Wu

whats2000mc@gmail.com

1. Introduction:

The field of computer vision consistently seeks efficient object detection methods. This study explores adapting a pre-trained ResNet model within the YOLO framework, focusing on developing a compatible loss function to enhance object localization and classification accuracy.

2. Methodology:

This study's methodology is centered around a custom loss function for the YOLO-based object detection model, comprising four main components:

A. **Class Prediction Loss (`get_class_prediction_loss`):**

- **Purpose**: This function is crucial for accurate class prediction of each detected object.
- **Method**: It applies a mask to focus on predictions in cells containing objects, using Mean Squared Error (MSE) to ensure precise class probability distribution.
- **Detail**: The loss is calculated only for those cells that actually contain objects, thereby enhancing classification accuracy.

B. **No-Object Loss (`get_no_object_loss`):**

- **Purpose**: Essential for accurately predicting the absence of objects in certain cells.
- **Method**: This function penalizes incorrect confidence predictions for cells that do not contain objects.
- **Detail**: It ensures that the model does not falsely detect objects where there are none, improving the overall reliability of the detection.

C. **Containment Confidence Loss (`get_contain_conf_loss`):**

- **Purpose**: This loss component refines the model's confidence in its predictions regarding object containment.
- **Method**: By applying MSE loss to the confidence scores of bounding boxes, it ensures that the model accurately gauges its certainty in the presence of objects.
- **Detail**: This function contributes significantly to reducing false positives and improving detection confidence.

`

D. **Regression Loss (`get_regression_loss`):**

- **Purpose**: Focused on the accuracy of bounding box coordinates.
- **Method**: Utilizes MSE loss for the center coordinates and dimensions of the bounding boxes.
- **Detail**: This function is pivotal for ensuring that the model precisely predicts the location and size of each detected object.

Customizable coefficients like `l_coord` (5) for bounding box regression and l_noobj (0.5) for no-object confidence predictions allow for nuanced calibration of the model's sensitivity to different detection aspects.

3. Results:

The adaptation of the YOLO model, using a pre-trained ResNet as the backbone, led to significant achievements in object detection. On the VOCdevkit_2007 validation set, the model reached a mean Average Precision (mAP) of 0.5605. When employing the Exponential Moving Average (EMA) technique, the model's performance was further enhanced, achieving a mAP of 0.5697. (Figure 2)

In the test set, comprising 4950 images, the model exhibited a mAP of 0.4295. This test set performance, while slightly lower than the validation results, still underscores the model's effectiveness in generalizing to new, unseen data.

4. Analysis & Interpretation

The experiment demonstrates the model's strong capability in object detection, particularly when trained on a well-structured dataset. The EMA model's superior performance in the validation set is indicative of the benefits that come with more stable and consistent training methodologies.

The slight drop in mAP on the test set compared to the validation set could indicate areas for improvement in model generalization. This discrepancy also highlights the importance of considering diverse datasets and scenarios in training to enhance the model's robustness and applicability to real-world scenarios.

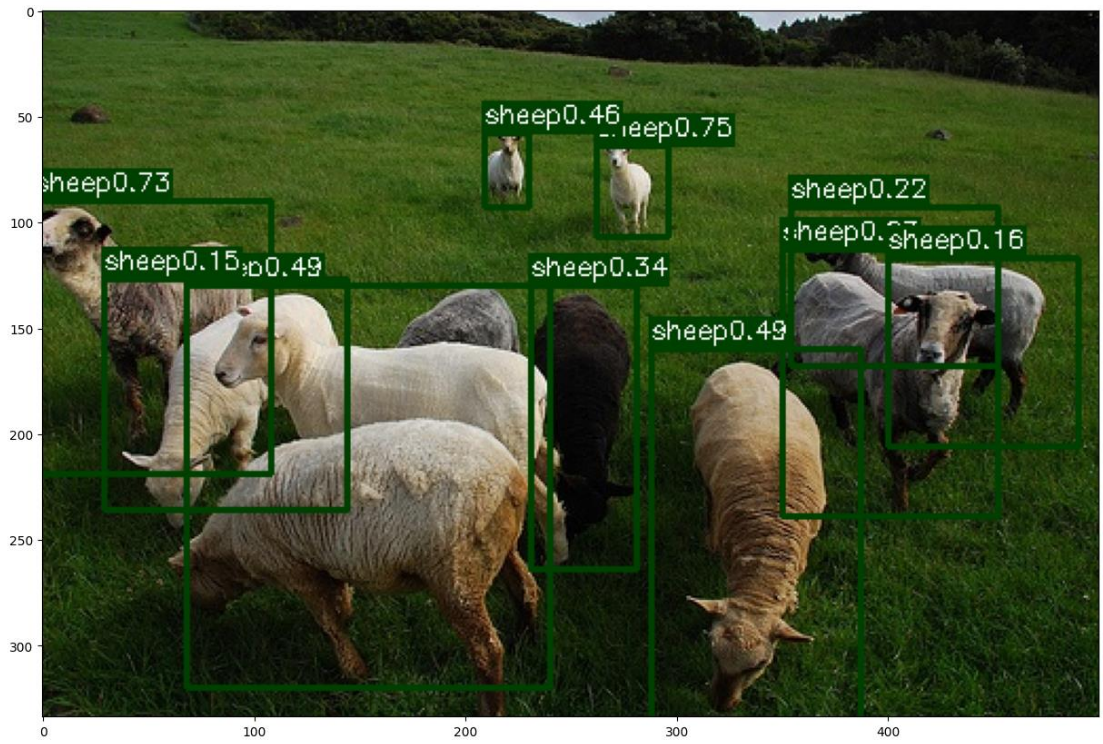
5. Conclusion

This study successfully demonstrates the effectiveness of leveraging pre-trained networks within the YOLO framework, enhanced by advanced training techniques like EMA. The achieved mAPs, both on the validation and test sets, establish the model as a robust tool for object detection tasks. Future work will focus on further improving the model's generalization capabilities and exploring additional advancements in loss functions and training strategies.

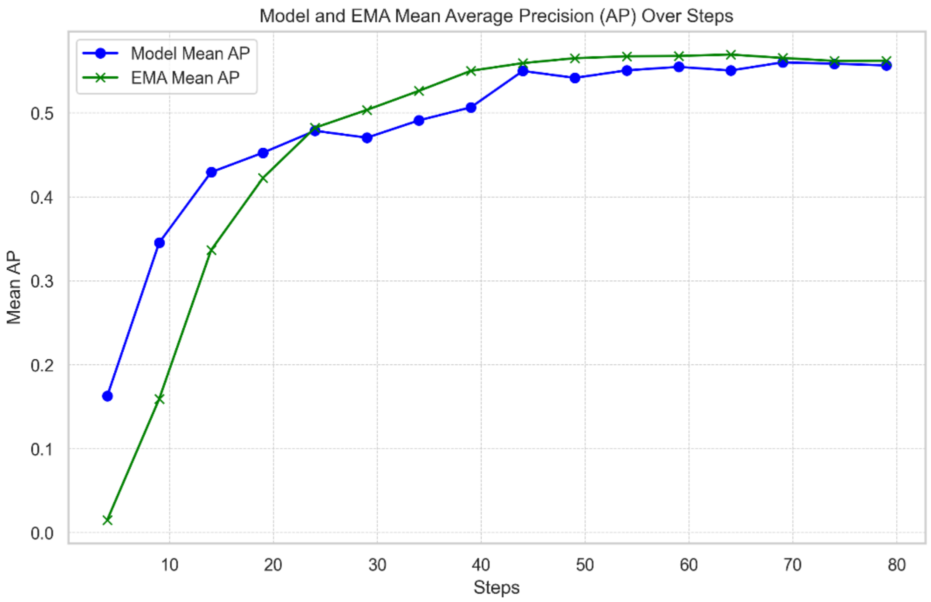
`

6. Appendices

This section provides additional visual evidence to complement the findings discussed in the report:



^ *Figure 1: Detected Images Examples*



^ *Figure 2: MAP overview*

`

Harnessing Pre-trained DarkNet53 for YOLO(V3)-Based Object Detection:
A Loss Function For multiple anchor heads Journey

1. Introduction:

This experiment evaluates an enhanced YOLOv3 model architecture with multiple anchor box heads and fixed-scale anchor boxes, contrasting with a previous approach featuring a single head with two boxes. The objective was to assess improvements in detection accuracy and generalization.

2. Methodology:

The Architecture is enhanced YOLOv3 with multiple anchor box heads.
We use `YoloLossV3` class with distinct components for object, no-object, box, and class loss, featuring specific lambda values for balancing. Below is the detail of that:

A. Classification Loss (`get_class_prediction_loss`):

- **Purpose**: Improves accuracy in classifying detected objects.
- **Method**: Cross-Entropy loss for cells with objects, comparing predicted class probabilities with actual labels. (This is different from the paper which use binary cross entropy)
- **Comparison**: Offers refined class discrimination compared to Experience 1's more basic approach, enhancing multi-class detection capabilities.

B. No-Object Loss (`get_no_object_loss`):

- **Purpose**: Accurate prediction of empty cells.
- **Method**: Binary Cross-Entropy loss applied to cells without objects.
- **Comparison**: This refined method decreases false positives, contrasting with Experience 1's simpler approach.

C. Object Loss (`get_object_loss`):

- **Purpose**: Enhances object detection confidence.
- **Method**: Utilizes Mean Squared Error (MSE) to compare the model's confidence in object detection (objectness score) with the IoU between predicted and actual bounding boxes. This direct comparison of IoU with confidence scores is a more focused approach compared to the indirect containment confidence loss used in YOLOv1.
- **Comparison**: YOLOv3's object loss directly ties the confidence of detecting an object to the accuracy of the bounding box. In contrast, YOLOv1's loss function lacks this direct correlation, leading to potential discrepancies in the confidence of object detection.

`

D. Box Coordination Loss (`get_box_coordination_loss`):

- **Purpose**: Ensures precise bounding box placement.
- **Method**: Implements MSE loss on the predicted bounding box coordinates, adjusting them based on the anchor box dimensions. This involves a comparison between the predicted and actual box coordinates, incorporating the anchor sizes for more precise
- **Comparison**: This method marks an advancement over YOLOv1's regression loss. In YOLOv1, the loss function for box coordinates did not account for anchor sizes, potentially leading to less accurate localization compared to YOLOv3's approach.

Customizable coefficients like ` lambda_noobj` (10) for no-object confidence predictions and ` lambda_box` (2.5) for box coordinate predictions. I have tried the lambda same as the YOLO V1 but it results in bad performance.

3. Results:

Validation Set: The model achieved a maximum mAP of 0.656197 at step 95 for the model and 0.663312 at step 60 for the EMA (Exponential Moving Average) version. (Figure 2)

Test Set: On applying the best validation model to the test set, a mAP of 0.49653 was achieved. (Figure 2)

4. Analysis & Interpretation

The enhanced architecture demonstrated notable improvements, especially in detecting smaller objects, leading to a higher mean Average Precision (mAP).

This model also showed an increased tendency to generate multiple bounding boxes around the same object.

The EMA model continued to outperform, indicating robust learning and better generalization.

However, a notable challenge was the increase in processing time for Non-Maximum Suppression (NMS), particularly at lower thresholds, due to the significant rise in the number of bounding boxes generated, this result in increasement training time.

5. Conclusion

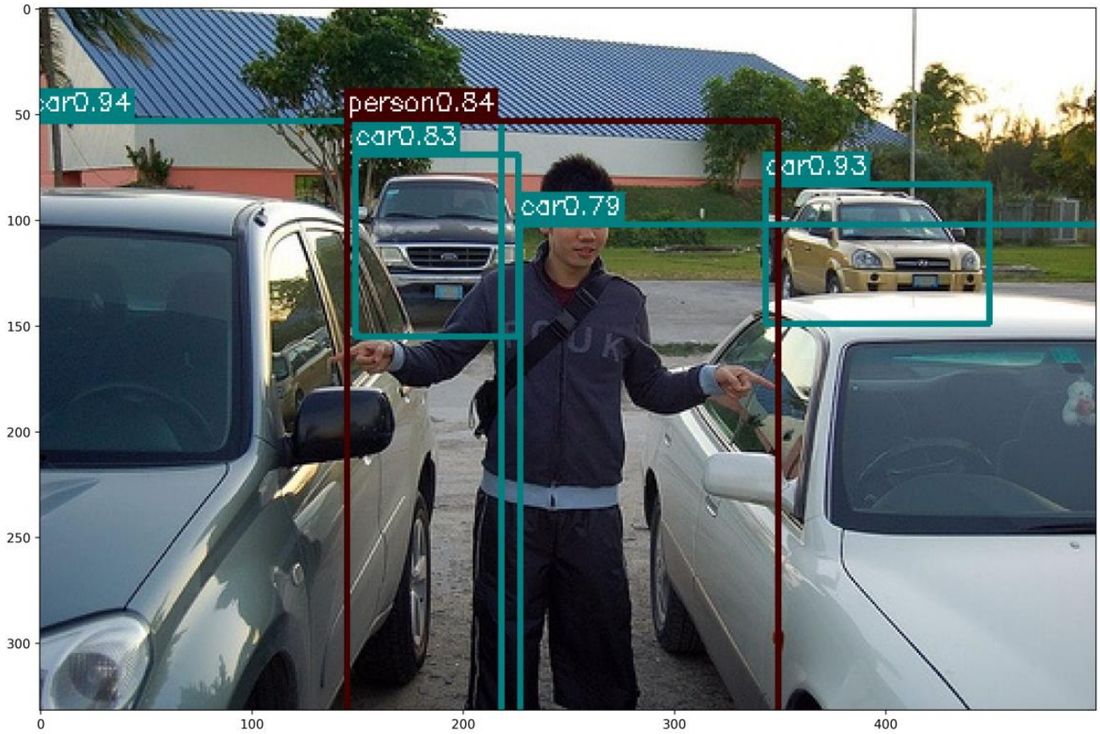
The model with Darknet53 and FPN demonstrates robust detection capabilities, as evidenced by its validation performance. However, the lower test mAP highlights the need for further tuning or data augmentation strategies to improve generalization to unseen data.

6. Appendices

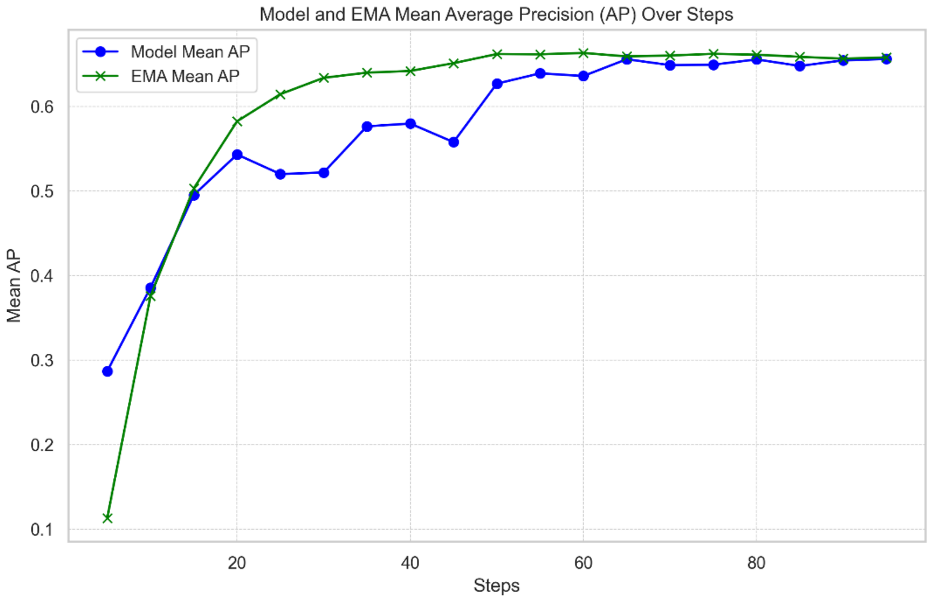
This section provides additional visual evidence to complement the findings

`

discussed in the report:



^ Figure 1: Detected Images Examples



^ Figure 2: MAP overview

`

Testing the YOLO model on video footage

1. Introduction:

This report explores the application of the YOLO model, developed in Experiment 2, to a real-world scenario by analyzing video data.

The aim is to assess the model's effectiveness in detecting objects in a dynamic environment and to identify areas for improvement.

2. Methodology:

1. Description of the video acquisition process using yt-dlp.
2. Steps involved in converting video frames into images using OpenCV.
3. Process of feeding these images into the model for object detection.
4. Explanation of converting the model's output into a CSV file for frame-wise analysis.
5. Details on drawing bounding boxes on frames and reconstructing them into a video.
6. Process of integrating the original audio back into the video.

3. Results:

The YOLO model demonstrated proficiency in detecting people, particularly in standard poses such as standing. This indicates a strong ability to recognize human figures in typical scenarios.

However, the model exhibited a tendency to incorrectly identify certain objects, such as mistaking square windows as TV objects or confusing other items as bottles. (Figure 1)

A significant challenge was observed with non-standard human postures. For example, people who were jumping or not in an upright position were sometimes misclassified as birds or other entities. (Figure 2)

4. Analysis & Interpretation

- **Strengths in Standard Detection**: The model's ability to accurately detect people in common postures reflects robust training on standard human figures. This aspect is particularly valuable for applications in environments with typical human activities.

- **Limitations in Object Identification**: The misclassification of inanimate objects and incorrect detections in complex environments suggest a need for more diverse and challenging data in the training set. Including more varied scenarios and objects could improve the model's discernment.

`

- **Challenges with Unusual Postures**: The model's struggle with non-standard human postures indicates a potential gap in the training dataset. Incorporating images of people in various activities and positions could enhance its recognition capabilities in real-world scenarios.

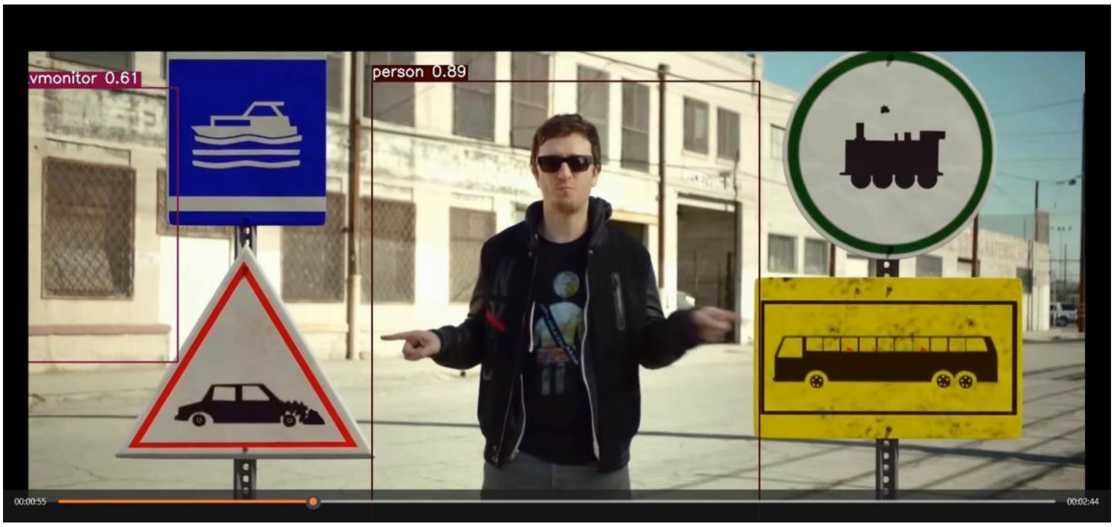
5. Conclusion

- **Effective yet Limited**: The model demonstrates effective detection capabilities, particularly for standard human postures and figures. However, its limitations in complex environments and with varied object shapes and human activities highlight areas for improvement.

- **Need for Diverse Training Data**: Incorporating a more diverse range of scenarios, objects, and human postures in the training data could significantly enhance the model's accuracy and applicability.

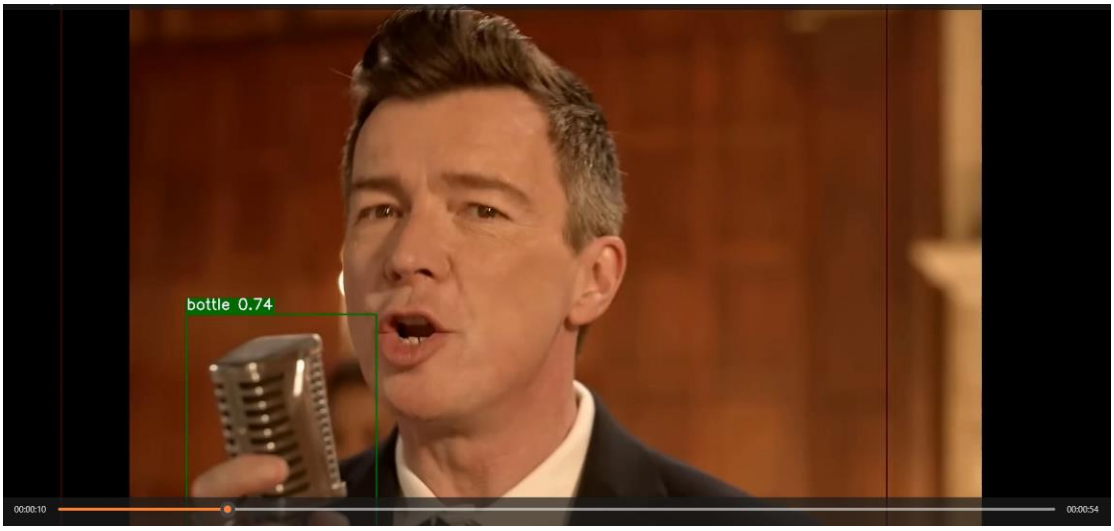
6. Appendices

This section provides additional visual evidence to complement the findings discussed in the report:

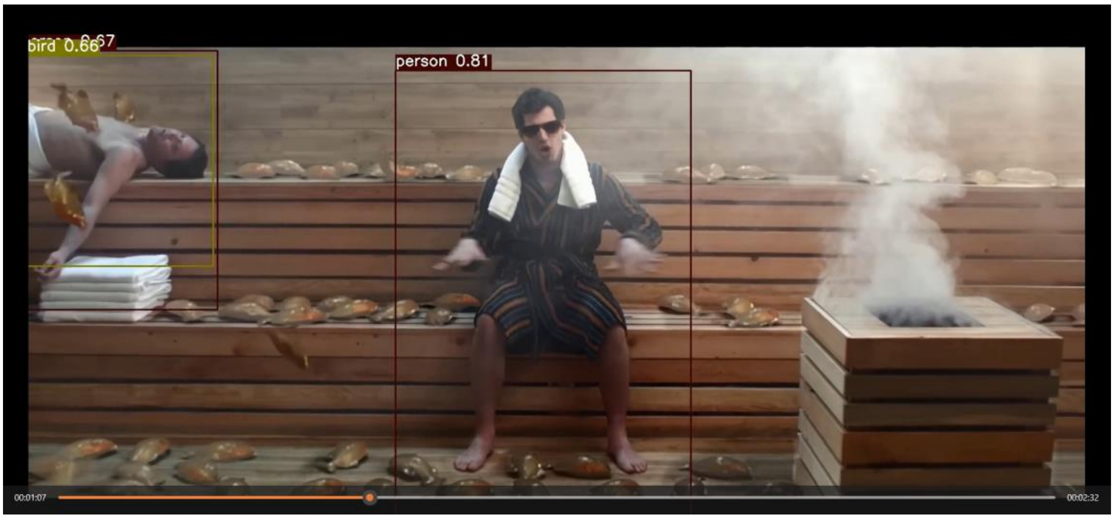


^ Figure 1a: Detected wrong in no object content, and on TV Monitor class

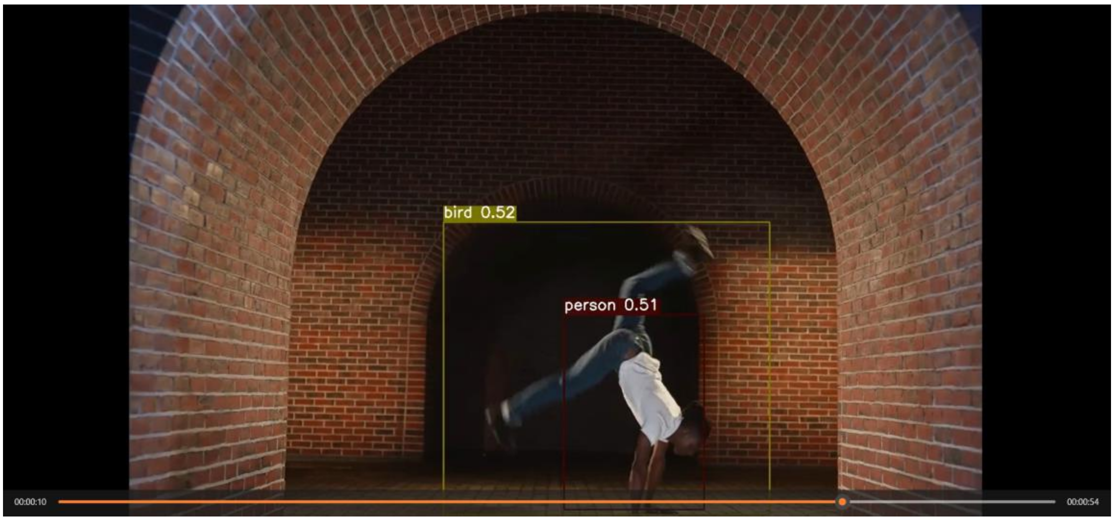
`



^ Figure 1b: Detected wrong in no object content, and on bottle class



^ Figure 2a: Detected in wrong class when the people is laying



^ Figure 2b: Detected in wrong class when the people is jumping

References

[1] OpenAI. (2023). ChatGPT [Large language model]. https://chat.openai.com

[2] YOLO V3 Implementation: https://github.com/BobLiu20/YOLOv3_PyTorch

[3] YOLO V3 Train Loop and Dataset modify from: https://github.com/aladdinpersson/Machine-Learning-Collection/blob/master/ML/Pytorch/object_detection/YOLOv3/

[4] Compare Between YOLO V1, V2, V3: Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3 | by Jonathan Hui | Medium

[5] Explain of the YOLO V1 Loss: neural networks - Yolo Loss function explanation - Cross Validated (stackexchange.com)

[6] YOLO V5 From Scratch: https://github.com/AlessandroMondin/YOLOV5m

[7] YOLO V3 Paper: https://arxiv.org/pdf/1804.02767.pdf

[8] YOLO V3 Loss: https://cloud.tencent.com/developer/article/1607664

[9] Video 1 for testing: https://youtu.be/xZGahvrep3o?si=qJcvQqSDYfMD8SBO

[10] Video 2 for testing: https://www.youtube.com/watch?v=GtL1huin9EE