Custom Object Detection and Novel Bounding Box Metric with YOLO

Setup:

- YOLO based object detection pipeline is implemented using ultralytics YOLOv5.
- Created a custom dataset with 300 images using **Cat and Dog** Kaggle dataset (https://www.kaggle.com/datasets/tongpython/cat-and-dog/data).
- Modifying the YOLOv5 training pipeline involves adjusting various aspects such as hyperparameters, data augmentation, model architecture, and loss functions to optimize performance. This includes fine-tuning learning rates, batch sizes, and momentum for better convergence, enhancing data augmentation techniques like flipping, scaling, and mosaic augmentation to improve generalization, and adjusting the model's depth and width for better feature extraction.
    - YOLOv5's hyperparameters can be adjusted in **train.py** when running the training script with arguments.
    - Data augmentations and the rest of the hyperparameters can be modified in **data/hyps/hyp.scratch-low.yaml** file.
    - For this task I used YOLOv5 – small model. Which suitable for small datasets. YOLOv5's model architecture is defined in **models/yolov5s.yaml**. Depth and width of the network can be changed according to the need using the **yolov5s.yaml** file.

Custom Bounding Box Similarity Metric:

- IoU is used as the default similarity metric between two bounding boxes.

$$IoU = \frac{Intersection\ Area}{Union\ Area}$$

- I propose a new similarity measure as a combination of *overlap area*, *aspect ratio*, *center alignment* and *scale* of the bounding boxes.

$$S_{area} = \frac{Intersection\ Area}{Union\ Area}$$

$$S_{center} = 1 - \frac{d}{d_{max}}$$

d - Euclidean distance between the bounding boxes centers

d_{max} - Diagonal length of the smallest box that encloses both bounding boxes

$$S_{aspect} = \frac{\min(r_1, r_2)}{\max(r_1, r_2)} \quad ; r_1\ (ratio\ of\ the\ box\ 1) = width/height$$

$$S_{scale} = \frac{\min(A_1, A_2)}{\max(A_1, A_2)} \quad A_1\,(area\ of\ the\ box\ 1) = width * height$$

All S_{area} , S_{center} , S_{aspect} and S_{scale} are normalized to be between [0, 1].

$$S_{Total} = \alpha_1 S_{area} + \alpha_2 S_{center} + \alpha_3 S_{aspect} + \alpha_4 S_{scale}$$

Here $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$

$$S_{Total} = 0.5\, S_{area} + 0.2\, S_{center} + 0.15\, S_{aspect} + 0.15\, S_{scale}$$

[Those α values can be treated as hyperparameters as well. For the simplicity for this task, I use manually selected values.]

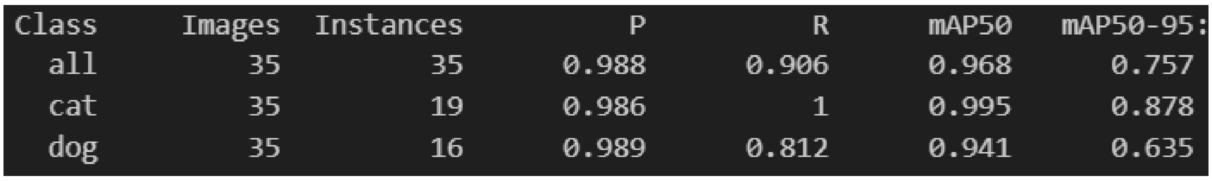
- Unlike IoU, which only measures the overlapping area between boxes, the above proposed metric considers center alignment, aspect ratio, and scale, offering a more comprehensive similarity evaluation. This can be beneficial in object detection tasks by ensuring that boxes not only overlap but also accurately match in position, shape, and size.

- This proposed metric is used as a post-training evaluation metric. Because in YOLOv5, CIoU is the default loss function. It is calculated based on the overlap area, aspect ratio, center alignment.

- This proposed metric can be integrated into evaluation script. In the *process_batch* (in evaluation script) , iou is calculated using *box_iou* function. It is possible to define a new function in *metric.py* script for the proposed metric and used that function to calculate the similarity metric value.

<u>Experimental Results and Analysis:</u>

When IoU as the similarity metric:



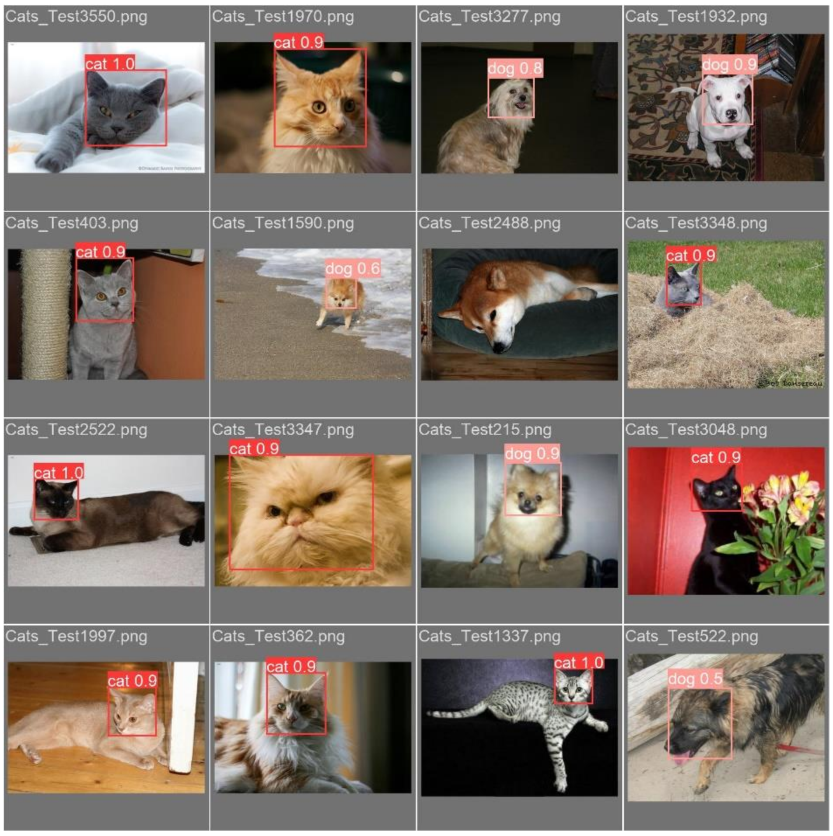
```
Class     Images  Instances          P          R      mAP50   mAP50-95:
  all         35         35      0.988      0.906      0.968      0.757
  cat         35         19      0.986          1      0.995      0.878
  dog         35         16      0.989      0.812      0.941      0.635
```

Precision: 0.988 (Very high; most detections are correct).

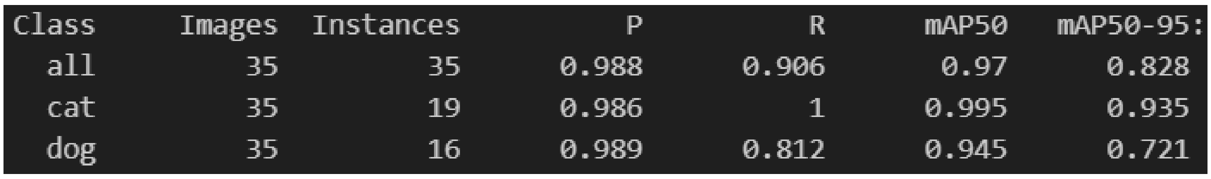
Recall: 0.906 (Very high; the model detects almost all objects).

mAP50: 0.968 (Very good detection accuracy at IoU=0.5).

mAP50-95: 0.757 (Decent performance across stricter IoU thresholds).



When the proposed Custom Bounding Box Similarity Metric as the similarity metric:



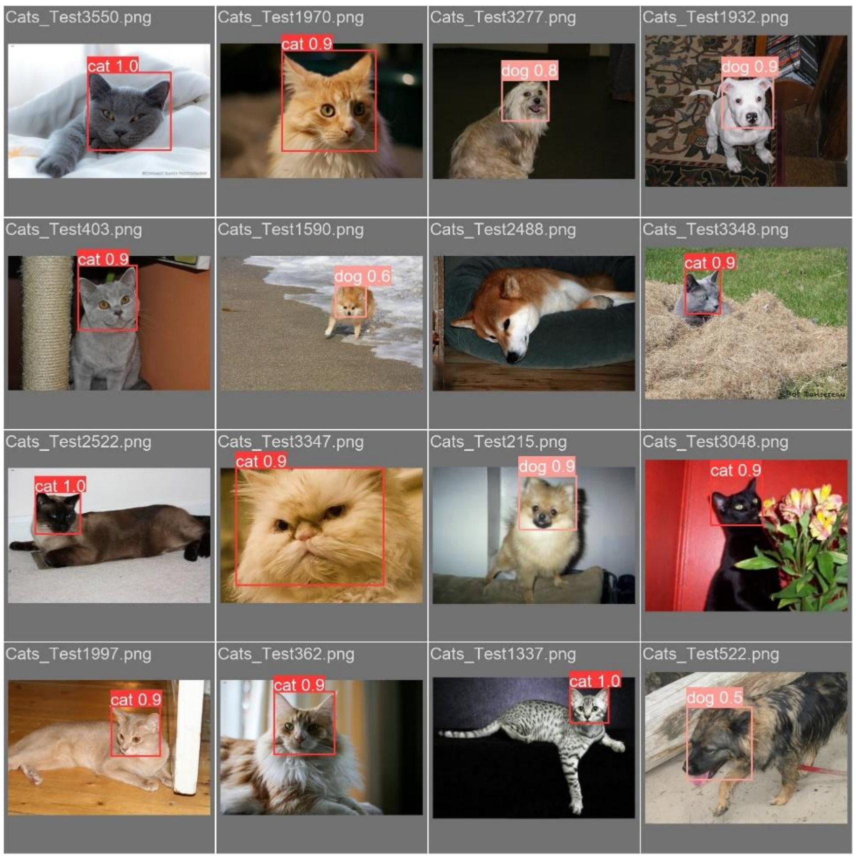
```
Class     Images  Instances          P          R      mAP50  mAP50-95:
  all         35         35      0.988      0.906       0.97      0.828
  cat         35         19      0.986          1      0.995      0.935
  dog         35         16      0.989      0.812      0.945      0.721
```

Precision: 0.988 (Very high; most detections are correct).

Recall: 0.906 (Very high; the model detects almost all objects).

mAP50: 0.97 (Very good detection accuracy at IoU=0.5).

mAP50-95: 0.828 (Good performance across stricter IoU thresholds).



- In both cases model is performing really well.
- Precision, Recall and mAP50 values for both similarity metrics are almost identical.
- But mAP50-95 values in the custom metric are higher than the IoU metric. It seems similarity calculations are more accurate when the custom metric is used for calculations.

There are more calculations in the proposed similarity metric calculation than IoU based metric calculation. Therefore, the proposed metric is more computationally expensive than IoU metric.

The proposed metric can be further improved by treating **α** values as hyperparameter during training. The best suitable values can be found in that way.