Abanoub Samir Girgis || 20190001
Joyce Fayek Milad || 20190160
Assignment 4
Group: S2

Assignment 4

Brief of Dataset

- The Common Object in Context (COCO) is one of the most popular large-scale labeled image datasets available for public use. It represents a handful of objects we encounter on a daily basis and contains image annotations in 80 categories, with over 1.5 million object instances. There are 5000 images for train, 128 images for validation and 17 images for test.

- The COCO dataset classes for object detection and tracking include the following pre-trained 80 objects:['person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear', 'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza', 'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed', 'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote', 'keyboard', 'cell phone', 'microwave', 'oven', 'toaster', 'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors', 'teddy bear', 'hair drier', 'toothbrush']

Summarize Yolo Architecture you used:

YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients, 16.5 GFLOPs

```
# Parameters
nc: 80  # number of classes
depth_multiple: 0.33  # model depth multiple
width_multiple: 0.50  # layer channel multiple
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32

# YOLOv5 v6.0 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 6, 2, 2]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
   [-1, 3, C3, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8
   [-1, 6, C3, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16
   [-1, 9, C3, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32
   [-1, 3, C3, [1024]],
   [-1, 1, SPPF, [1024, 5]],  # 9
  ]

# YOLOv5 v6.0 head
head:
  [[-1, 1, Conv, [512, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 6], 1, Concat, [1]],  # cat backbone P4
   [-1, 3, C3, [512, False]],  # 13

   [-1, 1, Conv, [256, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 4], 1, Concat, [1]],  # cat backbone P3
   [-1, 3, C3, [256, False]],  # 17 (P3/8-small)

   [-1, 1, Conv, [256, 3, 2]],
   [[-1, 14], 1, Concat, [1]],  # cat head P4
   [-1, 3, C3, [512, False]],  # 20 (P4/16-medium)

   [-1, 1, Conv, [512, 3, 2]],
   [[-1, 10], 1, Concat, [1]],  # cat head P5
   [-1, 3, C3, [1024, False]],  # 23 (P5/32-large)

   [[17, 20, 23], 1, Detect, [nc, anchors]],  # Detect(P3, P4, P5)
  ]
```

Activation functions:

- Leaky ReLU: used in Conv and hidden layers.
- Sigmoid: Output Layer.

Optimizer:

- SGD

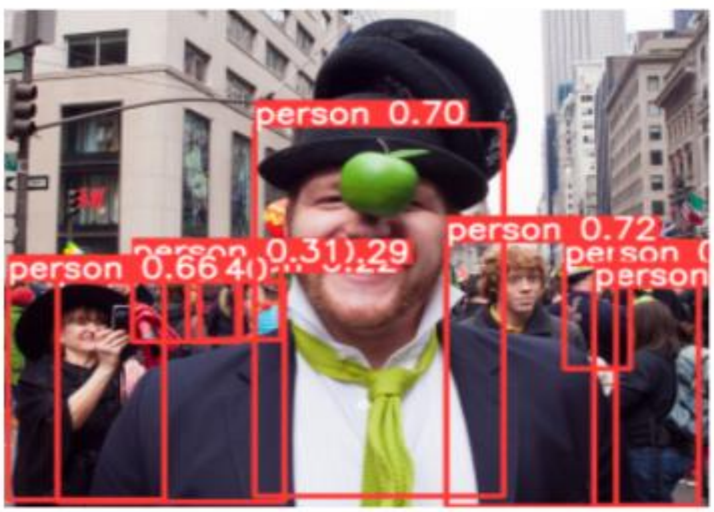
additional parameters

- Batch: 20
- epochs: 40
- image size: 416

Which loss function you use and why

Binary Cross-Entropy with Logits Loss: Because it compares each of the predicted probabilities to actual class output which can be either 0 or 1. It then calculates the score that penalizes the probabilities based on the distance from the expected value. That means how close or far from the actual value.

Predicted images



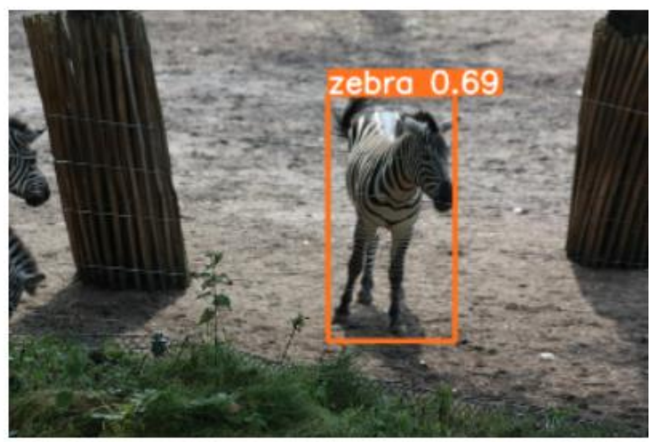

person 0.46
person 0.47
person 0.72


tv 0.34


toilet 0.31


tv 0.68


tv 0.38


zebra 0.69

Time for prediction

```
Fusing layers...
YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients, 16.5 GFLOPs
image 1/17 /content/test/1.jpg: 352x416 1 person, Done. (0.017s)
image 2/17 /content/test/10.jpg: 320x416 1 elephant, Done. (0.014s)
image 3/17 /content/test/11.jpg: 320x416 5 persons, 6 cars, Done. (0.009s)
image 4/17 /content/test/2.jpg: 288x416 4 persons, Done. (0.014s)
image 5/17 /content/test/3.jpg: 288x416 1 zebra, Done. (0.009s)
image 6/17 /content/test/5.jpg: 352x416 9 persons, Done. (0.012s)
image 7/17 /content/test/6.jpg: 416x288 4 persons, Done. (0.015s)
image 8/17 /content/test/7.jpg: 256x416 1 person, Done. (0.015s)
image 9/17 /content/test/8.jpg: 288x416 1 airplane, 1 kite, Done. (0.011s)
image 10/17 /content/test/airplane.jpg: 288x416 1 airplane, Done. (0.009s)
image 11/17 /content/test/car.jpg: 288x416 1 person, 19 cars, 1 motorcycle, Done. (0.010s)
image 12/17 /content/test/oranges(1).jpg: 288x416 2 bowls, 30 oranges, Done. (0.010s)
image 13/17 /content/test/toilet.jpg: 416x416 1 toilet, Done. (0.012s)
image 14/17 /content/test/train.jpg: 288x416 2 trains, 1 traffic light, Done. (0.010s)
image 15/17 /content/test/tv(1).jpg: 416x416 1 tv, Done. (0.012s)
image 16/17 /content/test/tv20.jpg: 416x416 1 tv, Done. (0.010s)
image 17/17 /content/test/tv21.jpg: 256x416 1 stop sign, 1 tv, 1 cell phone, Done. (0.010s)
Speed: 0.3ms pre-process, 11.7ms inference, 0.9ms NMS per image at shape (1, 3, 416, 416)
Results saved to runs/detect/exp11
```

Source Code:

https://colab.research.google.com/drive/1hJFQbEpT7iuzXSSZ3ZhM5FesnDAI6Uo8?usp=sharing

Use of custom dataset

Brief of Dataset (Dice Dataset)

- There are 75 images for train, 25 images for validation and 15 images for test, with 1-25 six-sided dice per image. Each image has an associated annotation txt file created by makesense.
- Classes: [‘1’, ‘2’, ‘3’, ‘4’, ‘5’, ‘6’]

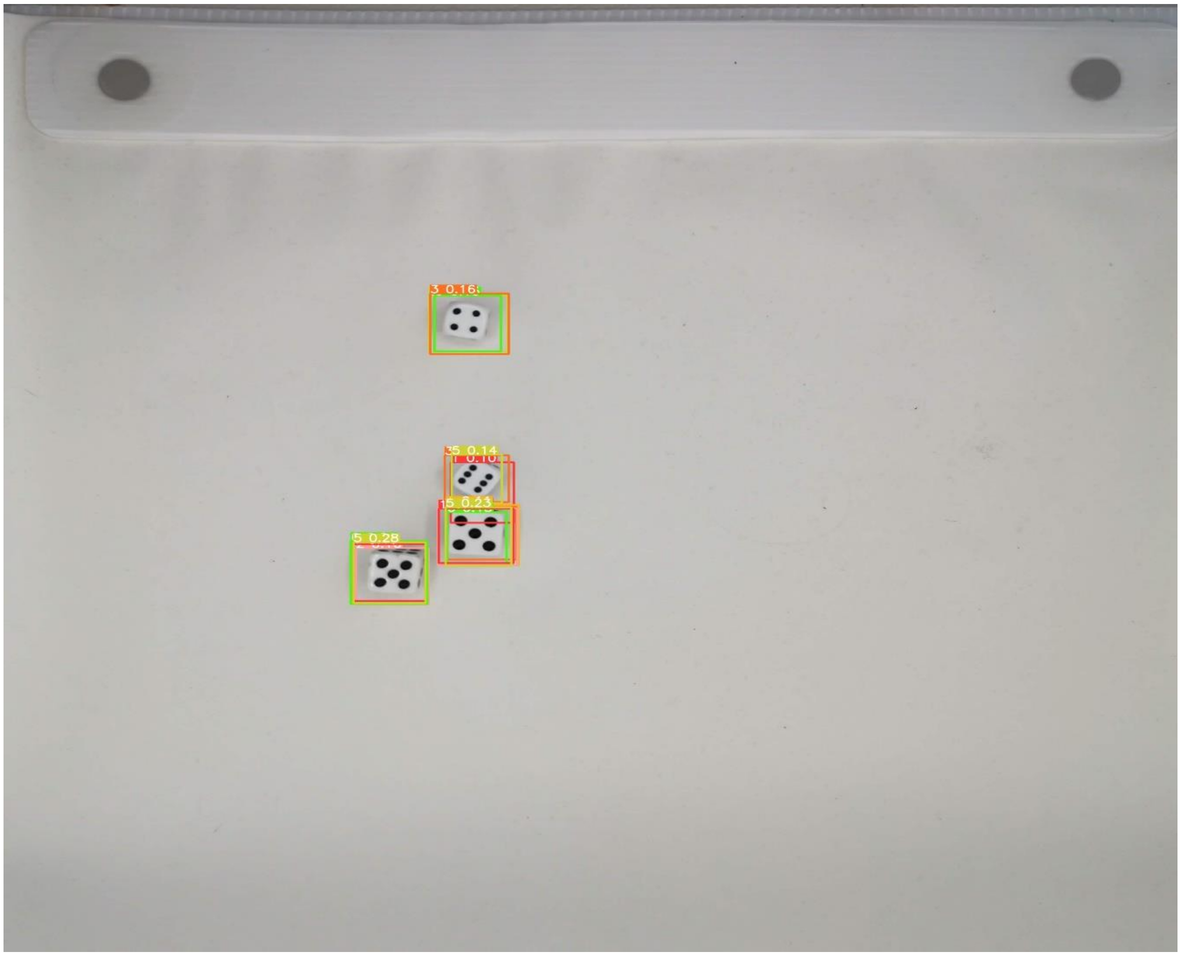
Summarize Yolo Architecture you used:

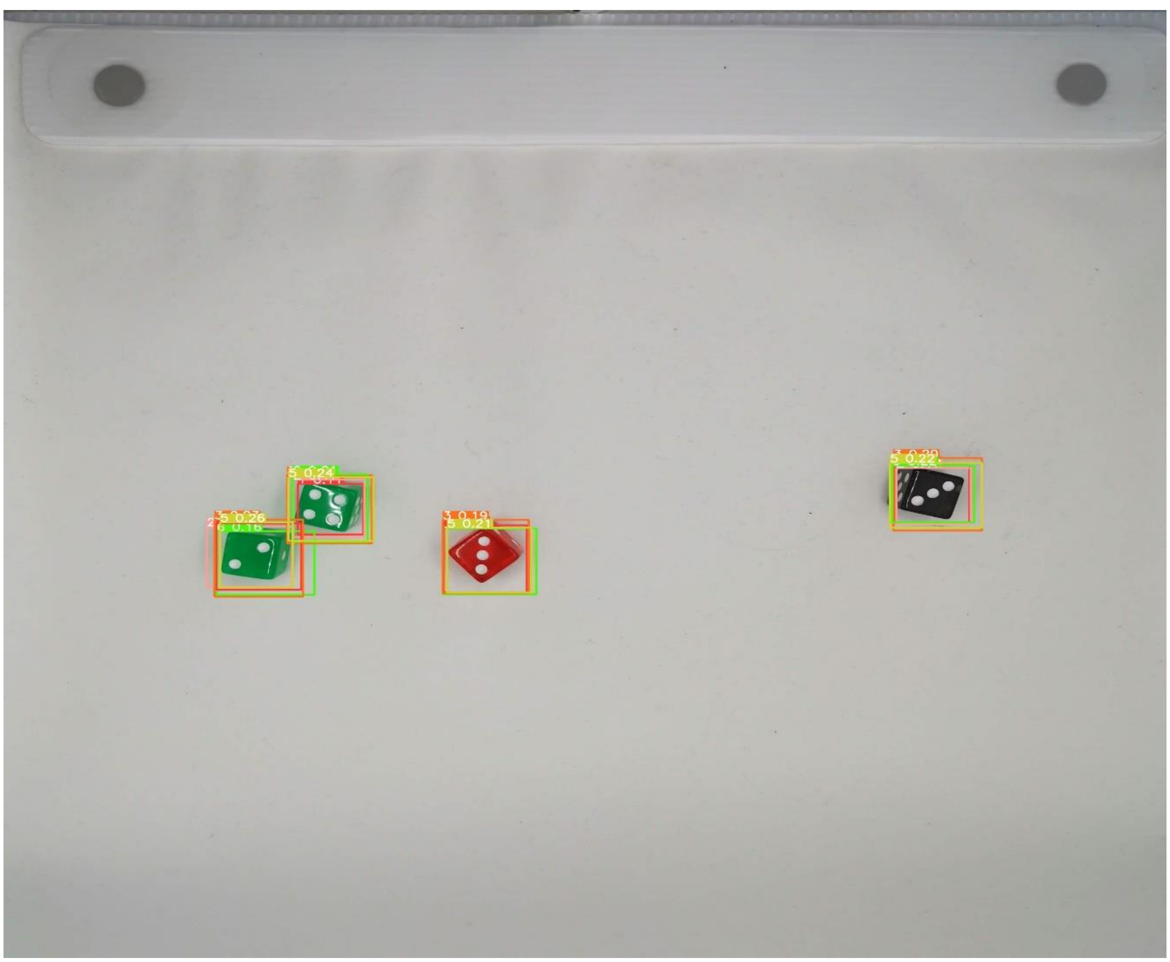
Same previous Architecture except (epochs = 75).

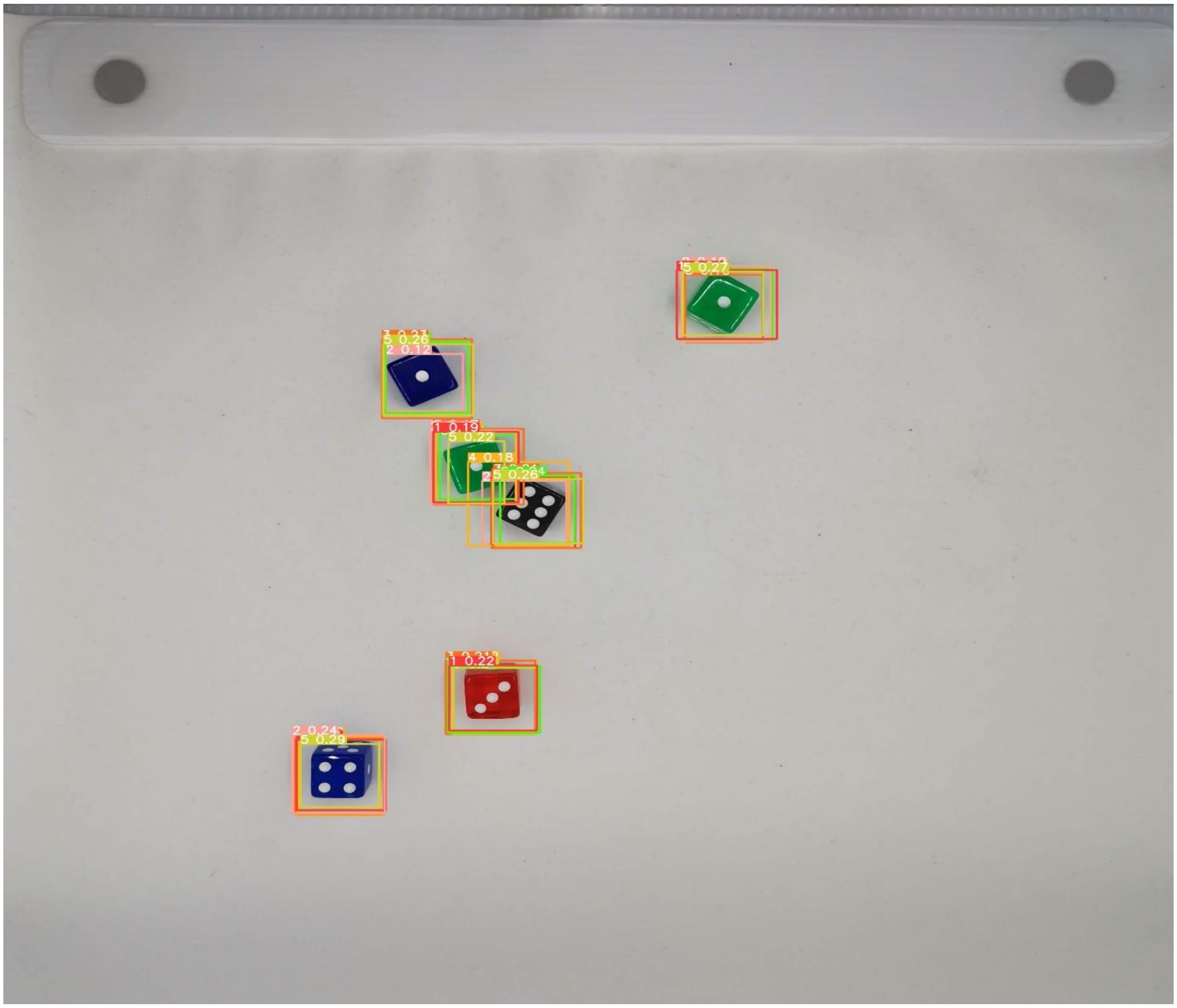
Which loss function you use and why

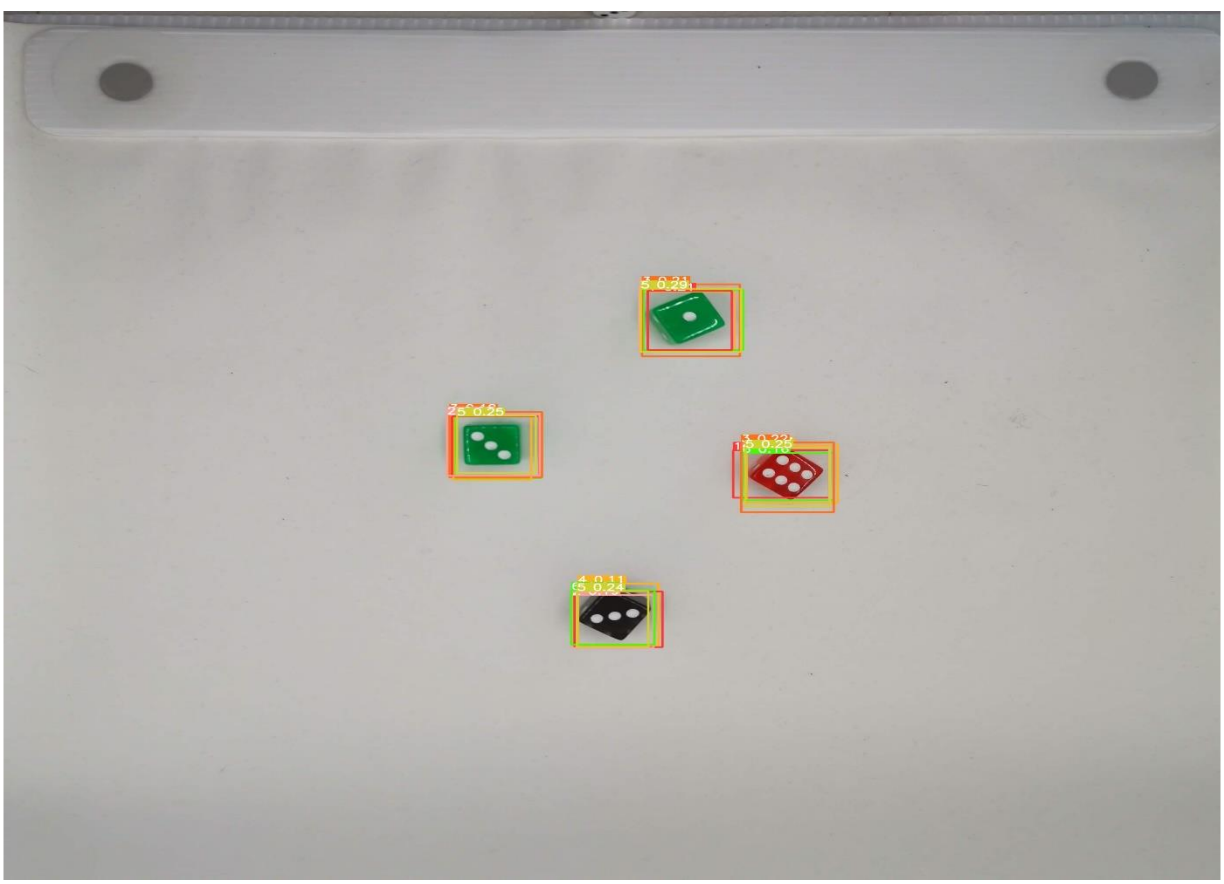
Same previous Loss Function

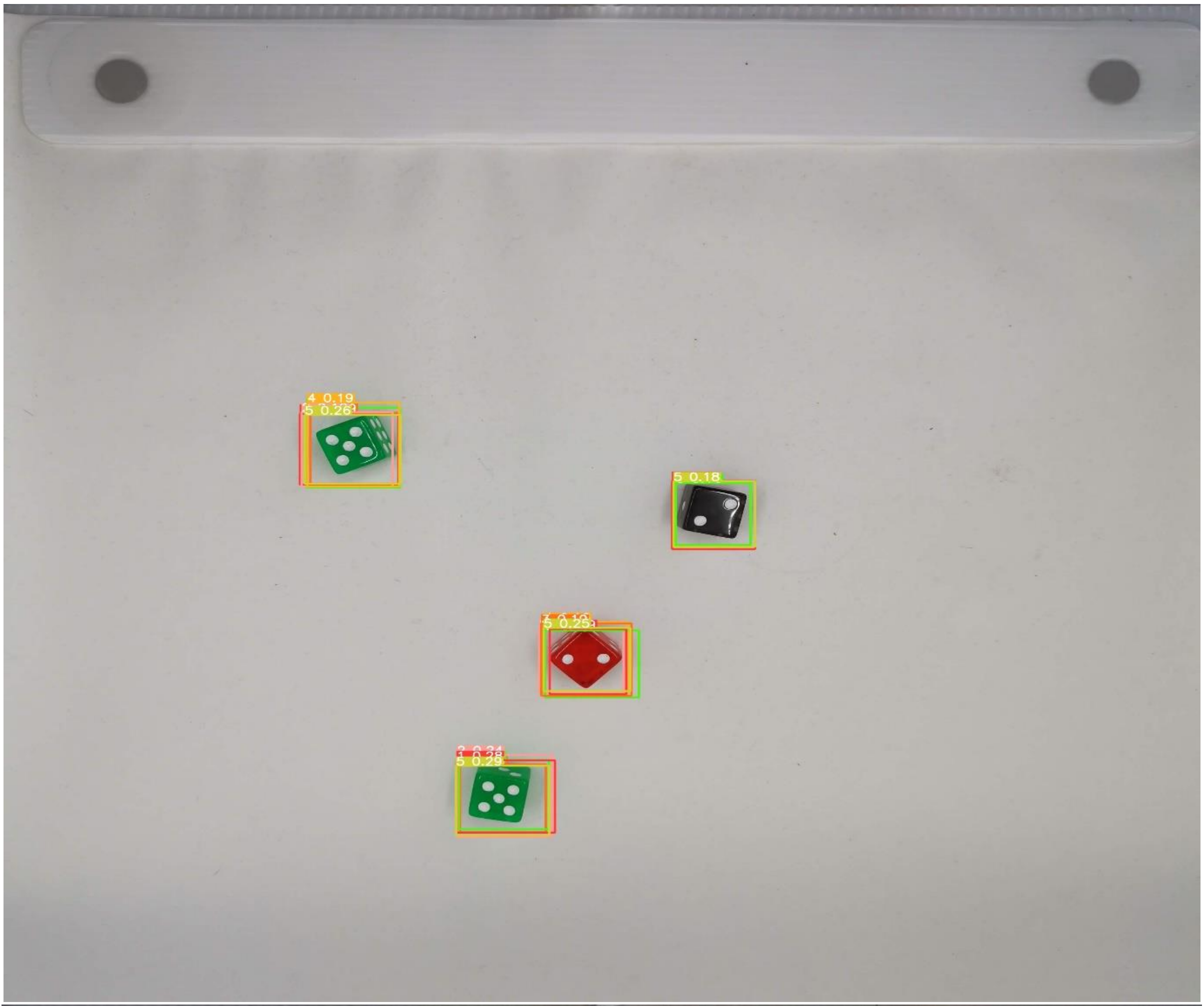
Predicted images

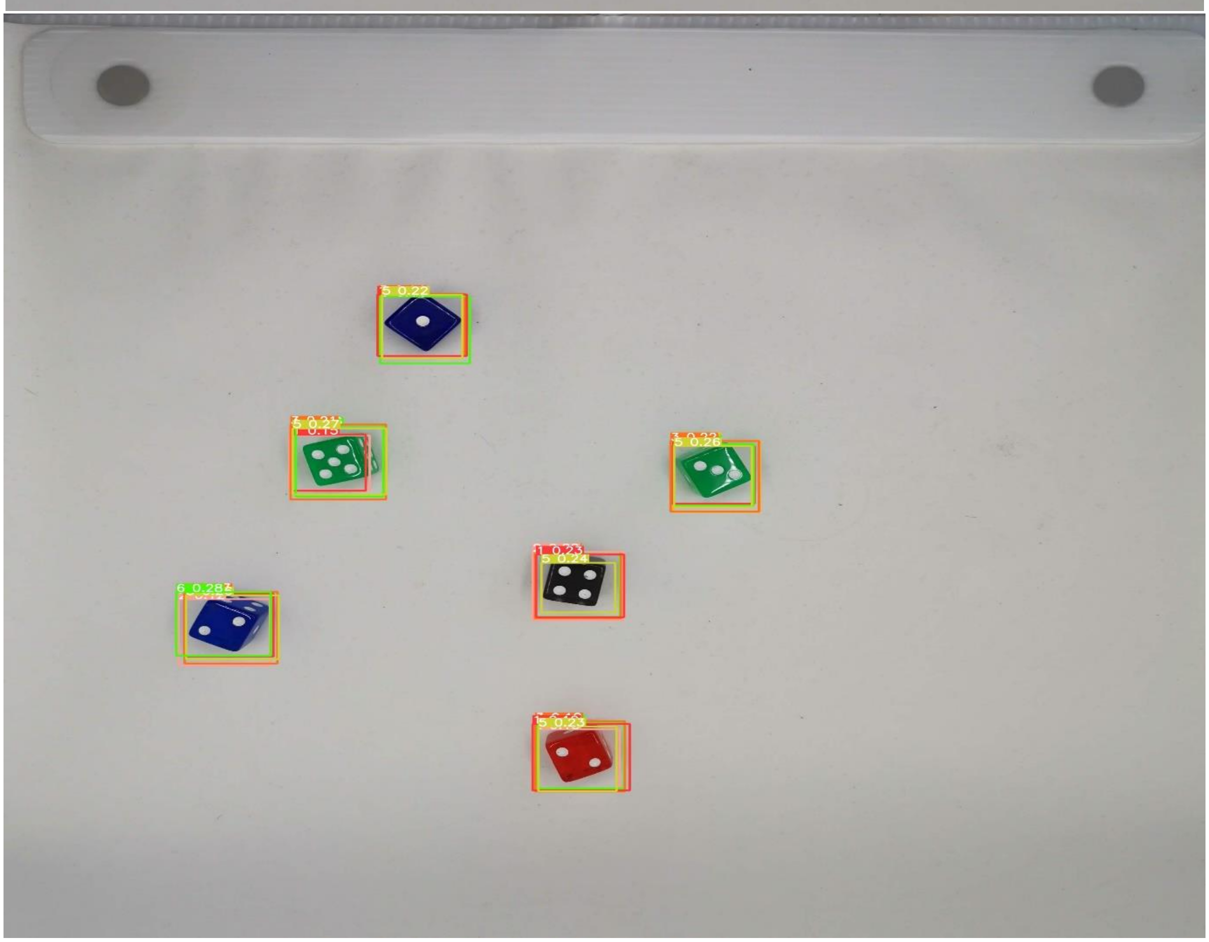


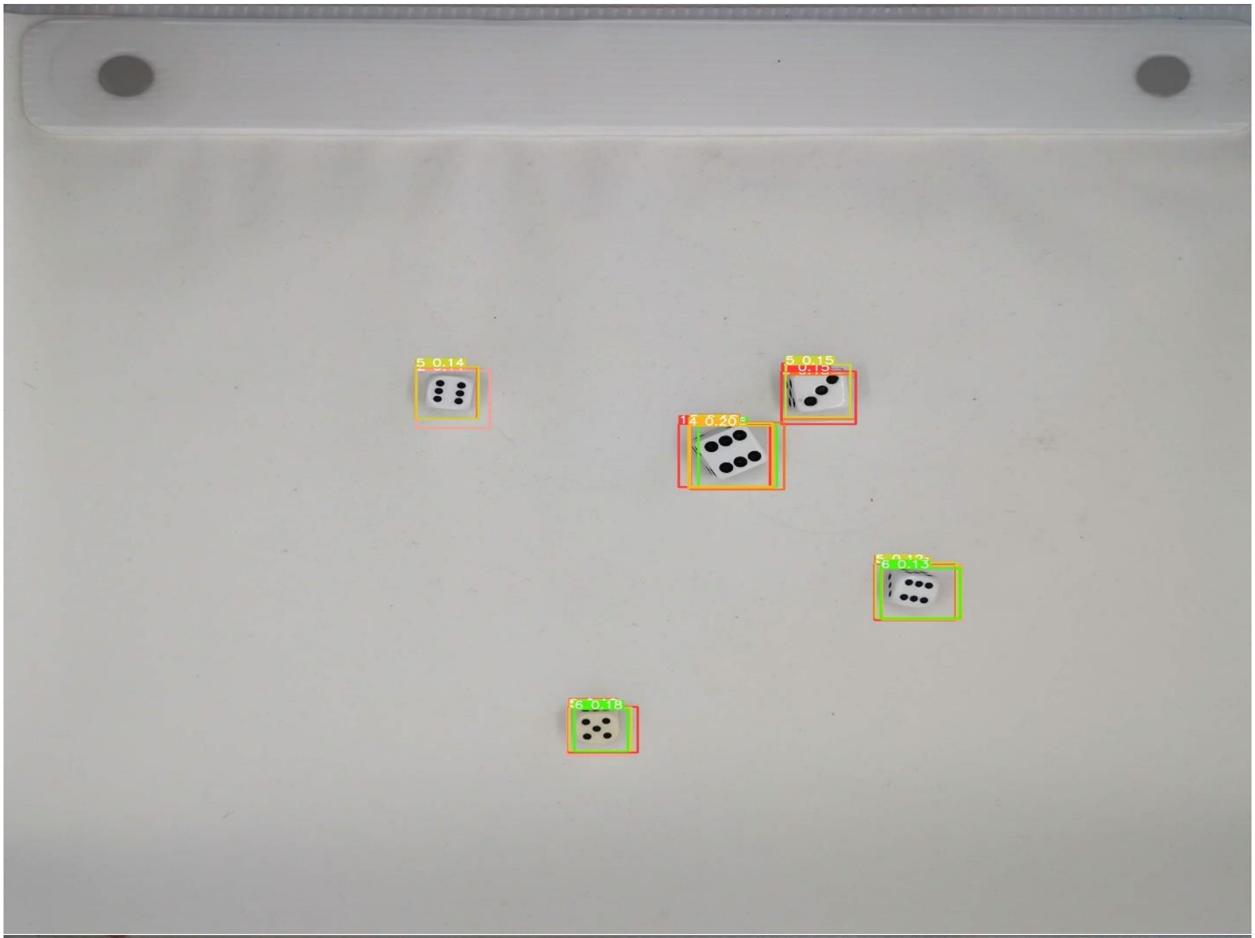


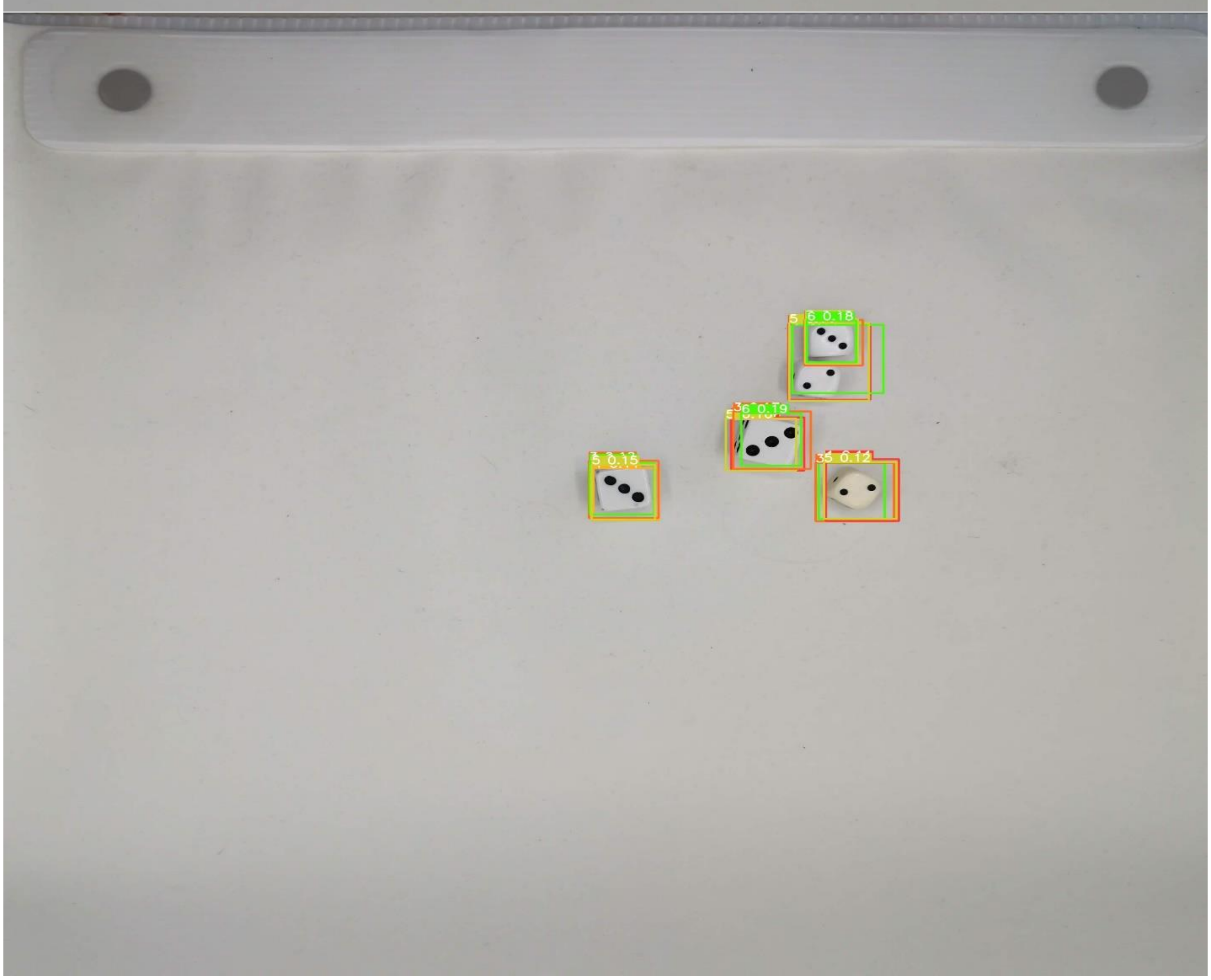


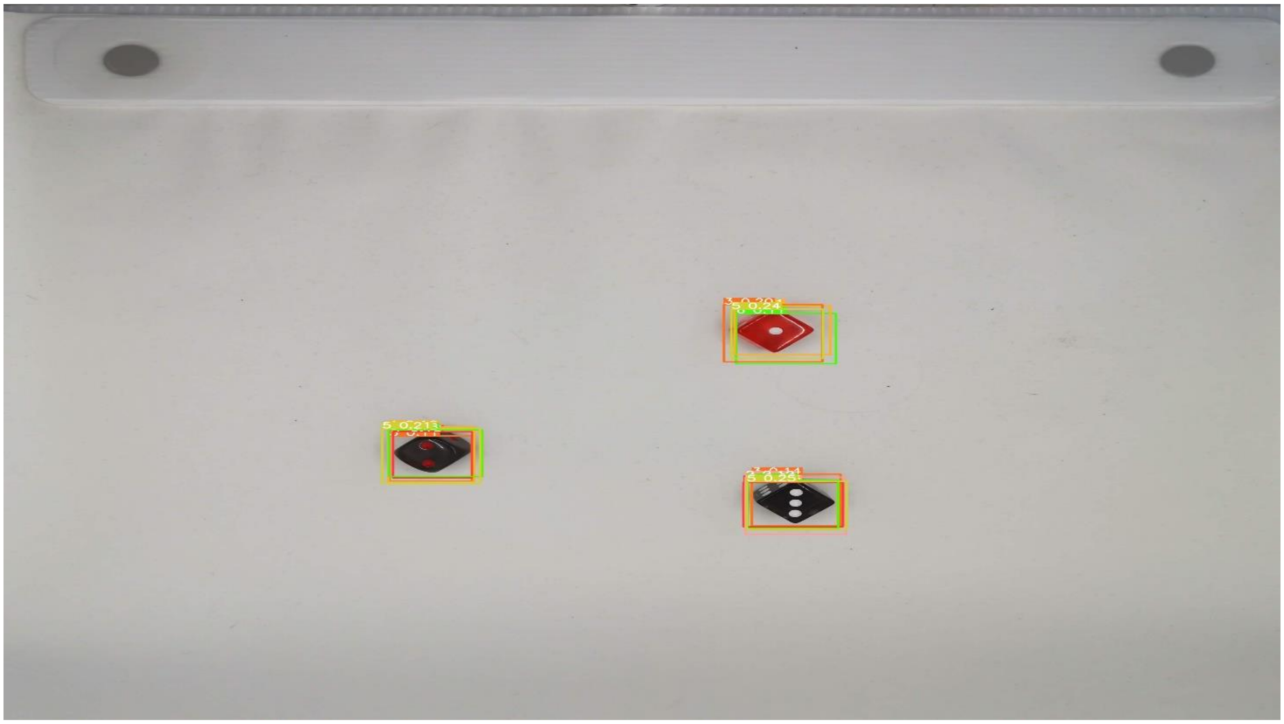


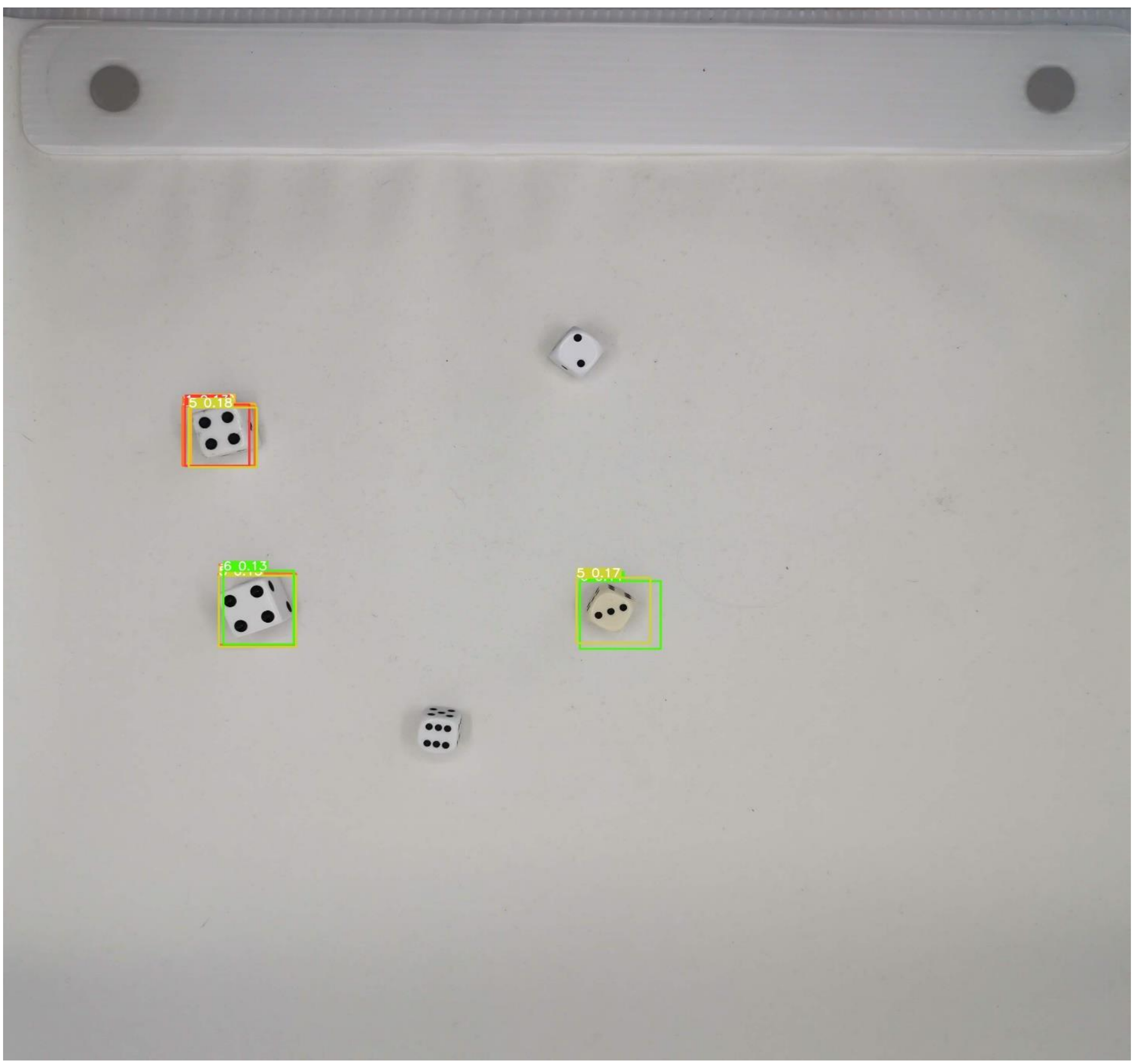

4 0.19
5 0.26
5 0.18
5 0.25
5 0.29

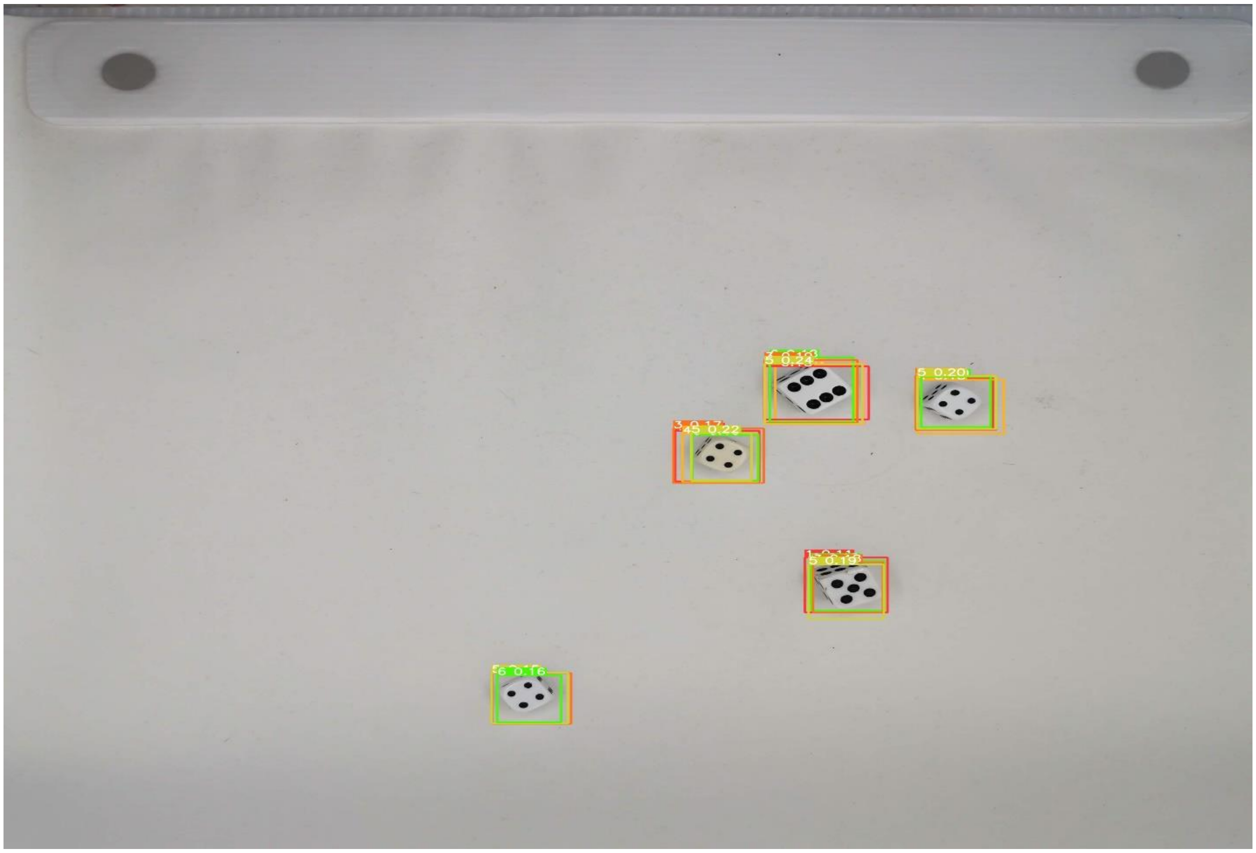

5 0.22
5 0.27
5 0.26
1 0.23
5 0.24
6 0.28
5 0.23

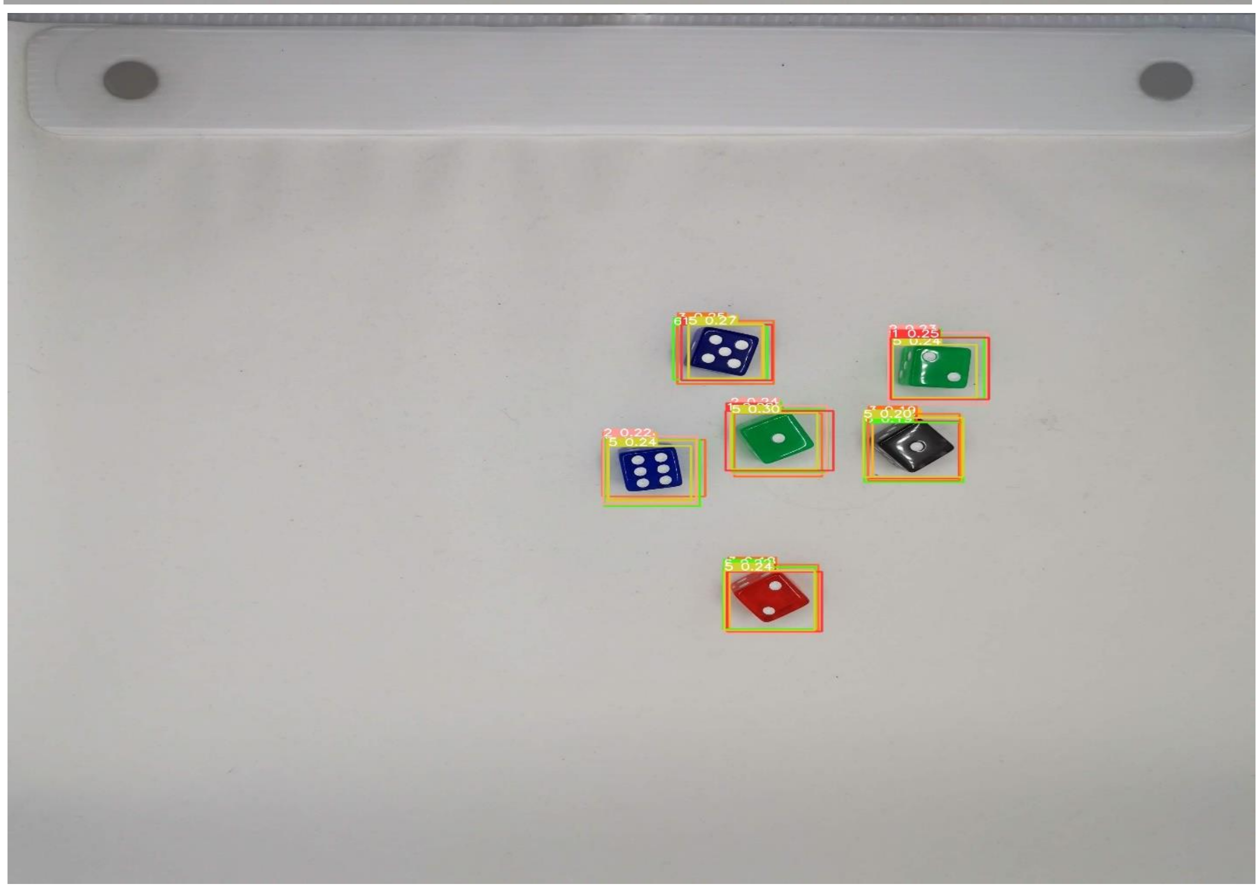





5 0.24


5 0.18
6 0.13
5 0.17


5 0.24
5 0.20
5 0.22
5 0.19
6 0.16


5 0.27
1 0.25
2 0.22
5 0.30
5 0.20
5 0.24

Time for prediction

```
Fusing layers...
YOLOv5s summary: 213 layers, 7026307 parameters, 0 gradients, 15.8 GFLOPs
image 1/15 /content/dice_dataset/test/IMG_20191209_100614.jpg: 416x416 2 1s, 2 2s, 3 3s, 3 4s, 3 5s, 3 6s, Done. (0.011s)
image 2/15 /content/dice_dataset/test/IMG_20191209_100620.jpg: 416x416 4 1s, 4 2s, 3 3s, 4 4s, 4 5s, 4 6s, Done. (0.011s)
image 3/15 /content/dice_dataset/test/IMG_20191209_100625.jpg: 416x416 4 1s, 3 2s, 4 3s, 1 4, 4 5s, 4 6s, Done. (0.011s)
image 4/15 /content/dice_dataset/test/IMG_20191209_100629.jpg: 416x416 4 1s, 3 2s, 3 3s, 2 4s, 4 5s, 4 6s, Done. (0.011s)
image 5/15 /content/dice_dataset/test/IMG_20191209_100633.jpg: 416x416 4 1s, 3 2s, 4 3s, 4 5s, 4 6s, Done. (0.012s)
image 6/15 /content/dice_dataset/test/IMG_20191209_100641.jpg: 416x416 5 1s, 4 2s, 5 3s, 4 4s, 6 5s, 6 6s, Done. (0.012s)
image 7/15 /content/dice_dataset/test/IMG_20191209_100645.jpg: 416x416 6 1s, 4 2s, 6 3s, 1 4, 6 5s, 6 6s, Done. (0.011s)
image 8/15 /content/dice_dataset/test/IMG_20191209_100649.jpg: 416x416 5 1s, 5 2s, 6 3s, 4 4s, 6 5s, 6 6s, Done. (0.011s)
image 9/15 /content/dice_dataset/test/IMG_20191209_100701.jpg: 416x416 3 1s, 2 3s, 3 5s, 3 6s, Done. (0.011s)
image 10/15 /content/dice_dataset/test/IMG_20191209_100706.jpg: 416x416 3 1s, 2 2s, 3 3s, 1 4, 4 5s, 3 6s, Done. (0.011s)
image 11/15 /content/dice_dataset/test/IMG_20191209_100714.jpg: 416x416 3 1s, 1 2, 4 3s, 1 4, 5 5s, 5 6s, Done. (0.011s)
image 12/15 /content/dice_dataset/test/IMG_20191209_100719.jpg: 416x416 4 1s, 3 2s, 4 3s, 2 4s, 4 5s, 4 6s, Done. (0.011s)
image 13/15 /content/dice_dataset/test/IMG_20191209_100722.jpg: 416x416 4 1s, 1 2, 5 3s, 3 4s, 5 5s, 5 6s, Done. (0.011s)
image 14/15 /content/dice_dataset/test/IMG_20191209_100728.jpg: 416x416 1 1, 2 3s, 1 4, 3 5s, 2 6s, Done. (0.011s)
image 15/15 /content/dice_dataset/test/IMG_20191209_100733.jpg: 416x416 4 1s, 3 2s, 3 3s, 2 4s, 5 5s, 3 6s, Done. (0.011s)
Speed: 0.5ms pre-process, 11.3ms inference, 1.1ms NMS per image at shape (1, 3, 416, 416)
Results saved to runs/detect/exp8
```

Source Code:

https://colab.research.google.com/drive/1enZdxisuAA0pj6OGGzPKEcTKUwGPwNJ9?usp=sharing