Laboratório 9 - Detecção de Objetos

Carlos R. A. Figueiredo[1]
Instituto Tecnológico de Aeronáutica, Laboratório de Inteligência Artificial para Robótica Móvel - CT-213. Professor Marcos Ricardo Omena de Albuquerque Máximo, São José dos Campos, São Paulo, 04 de junho de 2021.

[1]E-eletrônico: carlos.figueiredo@ga.ita.br

Para a primeira parte do laboratório, utilizando o Keras foi implementada a rede neural com as nove camadas, conforme pode ser visto na tabela 1:

Tabela 1. Sumário das camadas da rede.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 120, 160, 3)	0	
conv_1 (Conv2D)	(None, 120, 160, 8)	216	input_1[0][0]
norm_1 (BatchNormalization)	(None, 120, 160, 8)	32	conv_1[0][0]
leaky_relu_1 (LeakyReLU)	(None, 120, 160, 8)	0	norm_1[0][0]
conv_2 (Conv2D)	(None, 120, 160, 8)	576	leaky_relu_1[0][0]
norm_2 (BatchNormalization)	(None, 120, 160, 8)	32	conv_2[0][0]
leaky_relu_2 (LeakyReLU)	(None, 120, 160, 8)	0	norm_2[0][0]
conv_3 (Conv2D)	(None, 120, 160, 16)	1152	leaky_relu_2[0][0]
norm_3 (BatchNormalization)	(None, 120, 160, 16)	64	conv_3[0][0]
leaky_relu_3 (LeakyReLU)	(None, 120, 160, 16)	0	norm_3[0][0]
max_pool_3 (MaxPooling2D)	(None, 60, 80, 16)	0	leaky_relu_3[0][0]
conv_4 (Conv2D)	(None, 60, 80, 32)	4608	max_pool_3[0][0]
norm_4 (BatchNormalization)	(None, 60, 80, 32)	128	conv_4[0][0]
leaky_relu_4 (LeakyReLU)	(None, 60, 80, 32)	0	norm_4[0][0]
max_pool_4 (MaxPooling2D)	(None, 30, 40, 32)	0	leaky_relu_4[0][0]
conv_5 (Conv2D)	(None, 30, 40, 64)	18432	max_pool_4[0][0]
norm_5 (BatchNormalization)	(None, 30, 40, 64)	256	conv_5[0][0]
leaky_relu_5 (LeakyReLU)	(None, 30, 40, 64)	0	norm_5[0][0]
max_pool_5 (MaxPooling2D)	(None, 15, 20, 64)	0	leaky_relu_5[0][0]
conv_6 (Conv2D)	(None, 15, 20, 64)	36864	max_pool_5[0][0]
norm_6 (BatchNormalization)	(None, 15, 20, 64)	256	conv_6[0][0]
leaky_relu_6 (LeakyReLU)	(None, 15, 20, 64)	0	norm_6[0][0]
max_pool_6 (MaxPooling2D)	(None, 15, 20, 64)	0	leaky_relu_6[0][0]
conv_7 (Conv2D)	(None, 15, 20, 128)	73728	max_pool_6[0][0]
norm_7 (BatchNormalization)	(None, 15, 20, 128)	512	conv_7[0][0]

```
leaky_relu_7 (LeakyReLU)            (None, 15, 20, 128)  0             norm_7[0][0]
__________________________________________________________________________________________________
conv_skip (Conv2D)                  (None, 15, 20, 128)  8192          max_pool_6[0][0]
__________________________________________________________________________________________________
conv_8 (Conv2D)                     (None, 15, 20, 256)  294912        leaky_relu_7[0][0]
__________________________________________________________________________________________________
norm_skip (BatchNormalization)  (None, 15, 20, 128)  512       conv_skip[0][0]
__________________________________________________________________________________________________
norm_8 (BatchNormalization)         (None, 15, 20, 256)  1024          conv_8[0][0]
__________________________________________________________________________________________________
leaky_relu_skip (LeakyReLU)         (None, 15, 20, 128)  0             norm_skip[0][0]
__________________________________________________________________________________________________
leaky_relu_8 (LeakyReLU)            (None, 15, 20, 256)  0             norm_8[0][0]
__________________________________________________________________________________________________
concat (Concatenate)                (None, 15, 20, 384)  0             leaky_relu_skip[0][0]
                                                                       leaky_relu_8[0][0]
__________________________________________________________________________________________________
conv_9 (Conv2D)                     (None, 15, 20, 10)   3850          concat[0][0]
==================================================================================================
Total params: 445,346
Trainable params: 443,938
Non-trainable params: 1,408
__________________________________________________________________________________________________
```

Para a segunda parte do laboratório foram implementadas as funções para detecção de objetos com YOLO, de acordo com as fórmulas fornecidas no roteiro.

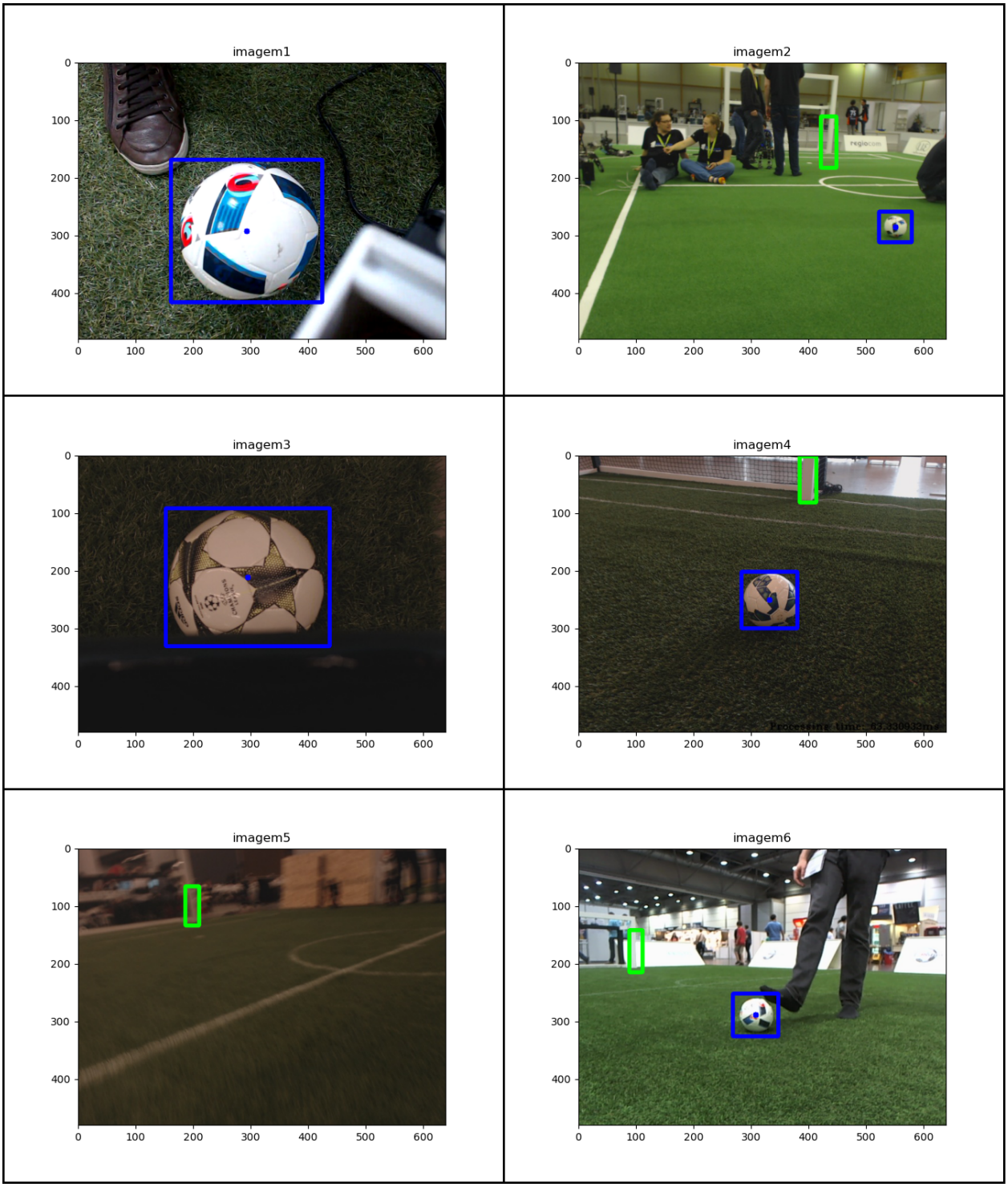
Para implementação da função `process_yolo_output()` em específico utilizou-se o seguinte algoritmo:

- Após definir variáveis das probabilidades e coordenadas, definiu-se a função sigmoid de output.
- Foram criados dois loops para buscar a maior probabilidade e coordenadas para a bola, para a trave1 e a segunda maior probabilidade para a trave2.
- Após isso é feita a normalização dos valores encontrados para definir as coordenadas e o tamanho dos três objetos pedidos.

Rodando o script `teste_vision_detector`, encontramos o seguinte resultado para 10 imagens selecionadas:

```
imagem1: [(ball_prob: 0.9999995, post1_prob: 0.00019692254, post2_prob: 0.00015154874), processing time:
0.2317962646484375]
imagem2: [(ball_prob: 0.99833494, post1_prob: 0.99965715, post2_prob: 0.009209948591887951), processing time:
0.03776144981384277]
imagem3: [(ball_prob: 0.9999894, post1_prob: 0.00182392, post2_prob: 0.0015491756), processing time:
0.03819608688354492]
imagem4: [(ball_prob: 0.99999976, post1_prob: 0.9999968, post2_prob: 0.002323827939108014), processing time:
0.04687857627868652]
imagem5: [(ball_prob: 0.0022087125, post1_prob: 0.99996746, post2_prob: 0.0033257483), processing time:
0.03125643730163574]
imagem6: [(ball_prob: 0.9971328, post1_prob: 0.99981874, post2_prob: 0.001241348683834076), processing time:
0.042262792587280027]
imagem7: [(ball_prob: 1.0, post1_prob: 0.99999523, post2_prob: 0.9999889), processing time: 0.03820013999938965]
imagem8: [(ball_prob: 0.99999905, post1_prob: 0.9999988, post2_prob: 0.9999709), processing time:
0.037761688232421875]
imagem9: [(ball_prob: 0.9333313, post1_prob: 0.9999993, post2_prob: 0.96711594), processing time:
0.048174142837524414]
imagem10: [(ball_prob: 0.9999999, post1_prob: 0.0011316637, post2_prob: 0.0011164375), processing time:
0.04687762260437012]
```

A detecção obtida pode ser vista nas figuras a seguir:



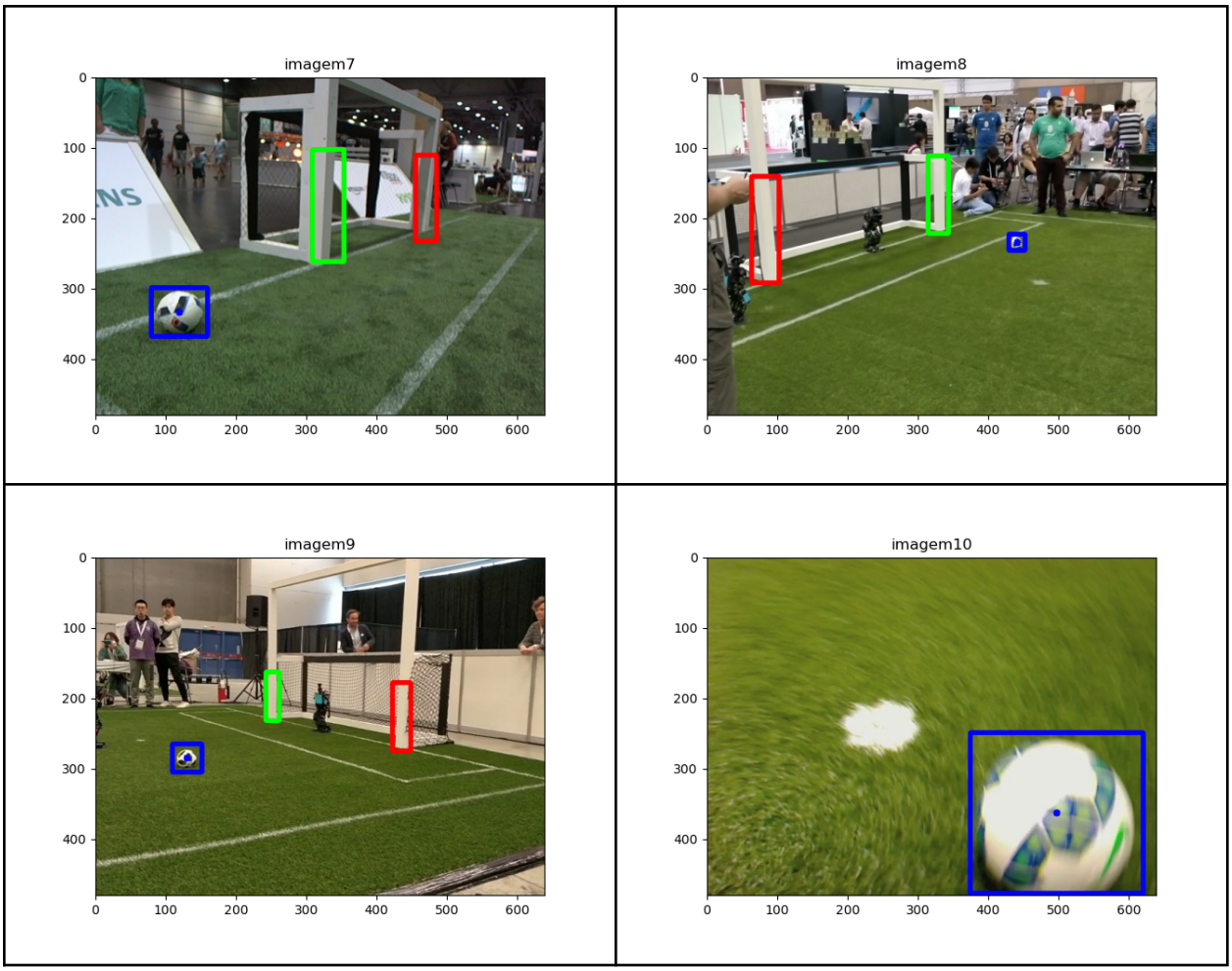


imagem7
imagem8
imagem9
imagem10
0
100
200
300
400
500
600