**FICHE - Apprentissage**

# 

**IO = Input et Output**

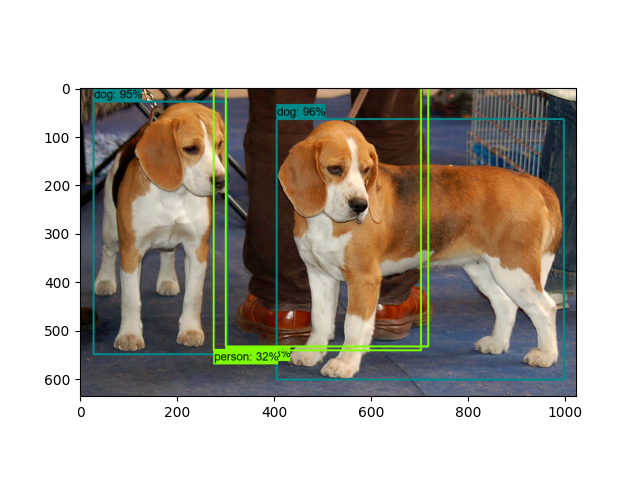
**INPUT**

Identifiez l'input précis (format informatique et mathématique) de l'algorithme et de chaque module. Donnez un exemple SVP !

Format XML, en 1D et utilisation de label map.

**OUTPUT**

Identifiez l'output précis de votre apprentissage. Donnez un exemple SVP !

Image Format JPG , 

**ORITHMENNÉ**

**OPTIMISATION**

Que tente d'optimiser votre algorithme, d'un point de vue données brutes (mathématique)

* Est-ce une erreur entre une valeur désirée et une valeur obtenue ? Si oui laquelle ?
* Est-ce une fonction d'évaluation est utilisée ? Si oui quelle est-elle ?|

TotalLoss prévu de minimum 2 et le mieux serait de 1 ou moins.

**ALGORITHME**

# ÉTAPES de l'ALGORITHME

**Décrivez votre algorithme en étapes détaillées (rétropropagation, évolution ou autre).**

1. **Extraction des feature**
2. **initialisation**
3. **Activation RELU\_ 6**
4. **Box coder**
5. **matcher**
6. **Box predictor**
7. **Anchor generator**
8. **Post processing**
9. **Analyse des loss**
10. **Training en tant que tel**

# ILLUSTRATION de l'ALGORITHME

Illustrez l'algorithme (couches convolutives, mutations, etc.)



**CODE**

# CONSTRUCTION

# Donnez le code de construction de votre structure d'apprentissage

# 

1 model {

2 ssd {

3 num\_classes: 1 *# Set this to the number of different label classes*

4 image\_resizer {

5 fixed\_shape\_resizer {

6 height: 640

7 width: 640

8 }

9 }

10 feature\_extractor {

11 type: "ssd\_resnet50\_v1\_fpn\_keras"

12 depth\_multiplier: 1.0

13 min\_depth: 16

14 conv\_hyperparams {

15 regularizer {

16 l2\_regularizer {

17 weight: 0.00039999998989515007

18 }

19 }

20 initializer {

21 truncated\_normal\_initializer {

22 mean: 0.0

23 stddev: 0.029999999329447746

24 }

25 }

26 activation: RELU\_6

27 batch\_norm {

28 decay: 0.996999979019165

29 scale: true

30 epsilon: 0.0010000000474974513

31 }

32 }

33 override\_base\_feature\_extractor\_hyperparams: true

34 fpn {

35 min\_level: 3

36 max\_level: 7

37 }

38 }

39 box\_coder {

40 faster\_rcnn\_box\_coder {

41 y\_scale: 10.0

42 x\_scale: 10.0

43 height\_scale: 5.0

44 width\_scale: 5.0

45 }

46 }

47 matcher {

48 argmax\_matcher {

49 matched\_threshold: 0.5

50 unmatched\_threshold: 0.5

51 ignore\_thresholds: false

52 negatives\_lower\_than\_unmatched: true

53 force\_match\_for\_each\_row: true

54 use\_matmul\_gather: true

55 }

56 }

57 similarity\_calculator {

58 iou\_similarity {

59 }

60 }

61 box\_predictor {

62 weight\_shared\_convolutional\_box\_predictor {

63 conv\_hyperparams {

64 regularizer {

65 l2\_regularizer {

66 weight: 0.00039999998989515007

67 }

68 }

69 initializer {

70 random\_normal\_initializer {

71 mean: 0.0

72 stddev: 0.009999999776482582

73 }

74 }

75 activation: RELU\_6

76 batch\_norm {

77 decay: 0.996999979019165

78 scale: true

79 epsilon: 0.0010000000474974513

80 }

81 }

82 depth: 256

83 num\_layers\_before\_predictor: 4

84 kernel\_size: 3

85 class\_prediction\_bias\_init: -4.599999904632568

86 }

87 }

88 anchor\_generator {

89 multiscale\_anchor\_generator {

90 min\_level: 3

91 max\_level: 7

92 anchor\_scale: 4.0

93 aspect\_ratios: 1.0

94 aspect\_ratios: 2.0

95 aspect\_ratios: 0.5

96 scales\_per\_octave: 2

97 }

98 }

99 post\_processing {

100 batch\_non\_max\_suppression {

101 score\_threshold: 9.99999993922529e-09

102 iou\_threshold: 0.6000000238418579

103 max\_detections\_per\_class: 100

104 max\_total\_detections: 100

105 use\_static\_shapes: false

106 }

107 score\_converter: SIGMOID

108 }

109 normalize\_loss\_by\_num\_matches: true

110 loss {

111 localization\_loss {

112 weighted\_smooth\_l1 {

113 }

114 }

115 classification\_loss {

116 weighted\_sigmoid\_focal {

117 gamma: 2.0

118 alpha: 0.25

119 }

120 }

121 classification\_weight: 1.0

122 localization\_weight: 1.0

123 }

124 encode\_background\_as\_zeros: true

125 normalize\_loc\_loss\_by\_codesize: true

126 inplace\_batchnorm\_update: true

127 freeze\_batchnorm: false

128 }

129}

130train\_config {

131 batch\_size: 8 *# Increase/Decrease this value depending on the available memory (Higher values require more memory and vice-versa)*

132 data\_augmentation\_options {

133 random\_horizontal\_flip {

134 }

135 }

136 data\_augmentation\_options {

137 random\_crop\_image {

138 min\_object\_covered: 0.0

139 min\_aspect\_ratio: 0.75

140 max\_aspect\_ratio: 3.0

141 min\_area: 0.75

142 max\_area: 1.0

143 overlap\_thresh: 0.0

144 }

145 }

146 sync\_replicas: true

147 optimizer {

148 momentum\_optimizer {

149 learning\_rate {

150 cosine\_decay\_learning\_rate {

151 learning\_rate\_base: 0.03999999910593033

152 total\_steps: 25000

153 warmup\_learning\_rate: 0.013333000242710114

154 warmup\_steps: 2000

155 }

156 }

157 momentum\_optimizer\_value: 0.8999999761581421

158 }

159 use\_moving\_average: false

160 }

161 fine\_tune\_checkpoint: "pre-trained-models/ssd\_resnet50\_v1\_fpn\_640x640\_coco17\_tpu-8/checkpoint/ckpt-0" *# Path to checkpoint of pre-trained model*

162 num\_steps: 25000

163 startup\_delay\_steps: 0.0

164 replicas\_to\_aggregate: 8

165 max\_number\_of\_boxes: 100

166 unpad\_groundtruth\_tensors: false

167 fine\_tune\_checkpoint\_type: "detection" *# Set this to "detection" since we want to be training the full detection model*

168 use\_bfloat16: false *# Set this to false if you are not training on a TPU*

169 fine\_tune\_checkpoint\_version: V2

170}

171train\_input\_reader {

172 label\_map\_path: "annotations/label\_map.pbtxt" *# Path to label map file*

173 tf\_record\_input\_reader {

174 input\_path: "annotations/train.record" *# Path to training TFRecord file*

175 }

176}

177eval\_config {

178 metrics\_set: "coco\_detection\_metrics"

179 use\_moving\_averages: false

180}

181eval\_input\_reader {

182 label\_map\_path: "annotations/label\_map.pbtxt" *# Path to label map file*

183 shuffle: false

184 num\_epochs: 1

185 tf\_record\_input\_reader {

186 input\_path: "annotations/test.record" *# Path to testing TFRecord*

187 }

188}

# 

# 

# EXÉCUTION

# Donnez le code qui lance l'exécution de l'apprentissage dans votre technologie

python model\_main\_tf2.py --model\_dir=models/my\_ssd\_resnet50\_v1\_fpn --pipeline\_config\_path=models/my\_ssd\_resnet50\_v1\_fpn/pipeline.config