



在 OpenVINO™ 精度检查器工具中添加 DIOU-NMS 指标，以获取 YOLO v4 的正确 mAP

白皮书

2021 年 1 月



您不得使用或方便他人使用本文档对此处描述的相关英特尔产品作任何侵权或其他法律分析。您同意就此后起草的任何专利权利（包括此处披露的主题）授予英特尔非排他性的免版税许可。

本文件不构成对任何知识产权的授权，包括明示的、暗示的，也无论是基于禁止反言的原则或其他。

此处提供的所有信息如有更改，恕不另行通知。请联系您的英特尔代表，了解最新的英特尔产品规格和路线图。

所述产品可能包含设计缺陷或错误（已在勘误表中注明），这可能会使产品偏离已经发布的技术规范。英特尔提供最新的勘误表备索。

如欲获取本文提及的带订购编号的文档副本，可致电 1-800-548-4725，或访问 www.intel.com/design/literature.htm。

英特尔技术的特性和优势取决于系统配置，可能需要支持的硬件、软件或服务激活。实际性能可能因系统配置的不同而有所差异。没有任何产品或组件是绝对安全的。请联系您的系统制造商或零售商，或访问 <http://www.intel.cn/content/www/cn/zh/homepage.html>。

没有任何产品或组件是绝对安全的。

英特尔和英特尔标志是英特尔公司或其子公司在美国和/或其他国家（地区）的商标。

* 其他的名称和品牌可能是其他所有者的资产。

© 2020 英特尔公司版权所有。

目录

1.0	简介.....	5
2.0	设置环境.....	6
3.0	添加 Diou-nms 的步骤	7
3.1.	安装精度检查器工具.....	7
3.2.	编辑 “/opt/intel/opencv_2021/deployment_tools/open_model_zoo/tools/accuracy_checker/accuracy_checker/postprocessor/nms.py”	7
4.0	用 Diou-NMS 检查 yolo-v4-tf 模型的精度.....	10
4.1.	下载和转换 yolo-v4-tf 模型	10
4.2.	准备数据集和 yml 文件	10
4.2.1.	下载数据集.....	10
4.2.2.	复制和编辑“accuracy-check.yml”	10
4.3.	运行精度检查器工具，计算 yolo-v4-tf 的统计精度.....	11
5.0	参考资料.....	12

修订记录

日期	修订版	说明
2021 年 1 月	0.8	初始版本。

§

1.0 简介

Yolov4 模型于 2020 年中期推出，它的问世对深度学习目标检测领域产生了深远影响。为取得成功，Yolov4 集成了许多一流的技术，其中包括改进的非极大值抑制 (NMS) 算法，该算法使用距离交并比 (DIoU) 而非交并比 (IoU)。DIoU 总结了边界框回归中的两个几何因素（即重叠区域和中心点距离），从而加快了收敛速度并提升了性能。

在本白皮书中，DIoU-NMS 函数将被添加到 OpenVINO™ 精度检查器工具中，以用于计算 yolov4 的正确预期精度。尽管 DIoU-NMS 可帮助大幅提升精度，但它也可以与训练后优化工具包 (pot) 一起使用，以生成优化的 INT8 模型。

2.0 设置环境

- 系统环境
 - 设置 OpenVINO™ 2021.2
- Ubuntu 18.04 或 Ubuntu 20.04。

§

3.0 添加 Diou-nms 的步骤

3.1. 安装精度检查器工具

按照 [openvinotoolkit open model zoo](#) 指南进行安装。

1. `cd /opt/intel/openvino_2021/deployment_tools/open_model_zoo/tools/accuracy_checker`
2. `sudo apt-get install python3 python3-dev python3-setuptools python3-pip`
3. `python3 setup.py install`

3.2. 编辑

`"/opt/intel/openvino_2021/deployment_tools/open_model_zoo/tools/accuracy_checker/accuracy_checker/postprocessor/nms.py"`

将以下代码添加到文件中：

```

1. #add DIOU-NMS support
2. class DIOU_NMS(Postprocessor):
3.     __provider = 'diou_nms'
4.
5.     prediction_types = (DetectionPrediction, ActionDetectionPrediction)
6.     annotation_types = (DetectionAnnotation, ActionDetectionAnnotation)
7.
8.     @classmethod
9.     def parameters(cls):
10.         parameters = super().parameters()
11.         parameters.update({
12.             'overlap':
13.                 NumberField(min_value=0, optional=True, default=0.5,
14.                             description="Overlap threshold for merging
15.                             detections."),
16.             'include_boundaries': BoolField(optional=True,
17.                                              default=True, description="Shows if boundaries are included."),
18.             'keep_top_k': NumberField(min_value=0, optional=True, description="Keep top K."),
19.             'use_min_area': BoolField(
20.                 optional=True, default=False,

```

在 OpenVINO 精度检查器工具中添加 Diou-NMS 指标，以获取 YOLO v4 的正确 mAP

白皮书

```

22.         description="Use minimum area of two boundi
ng boxes as base area to calculate overlap"
23.     )
24.     })
25.     return parameters
26.
27.     def configure(self):
28.         self.overlap = self.get_value_from_config('overlap'
)
29.         self.include_boundaries = self.get_value_from_conf
ig('include_boundaries')
30.         self.keep_top_k = self.get_value_from_config('keep_
top_k')
31.         self.use_min_area = self.get_value_from_config('use
_min_area')
32.
33.     def process_image(self, annotations, predictions):
34.         for prediction in predictions:
35.             scores = get_scores(prediction)
36.             keep = self.diou_nms(
37.                 prediction.x_mins, prediction.y_mins, predi
ction.x_maxs, prediction.y_maxs, scores,
38.                 self.overlap, self.include_boundaries, self
.keep_top_k, self.use_min_area
39.             )
40.             prediction.remove([box for box in range(len(pre
diction.x_mins)) if box not in keep])
41.
42.         return annotations, predictions
43.
44.     @staticmethod
45.     def diou_nms(x1, y1, x2, y2, scores, thresh, include_bo
undaries=True, keep_top_k=None, use_min_area=False):
46.         """
47.         Pure Python NMS baseline.
48.         """
49.         b = 1 if include_boundaries else 0
50.
51.         areas = (x2 - x1 + b) * (y2 - y1 + b)
52.         order = scores.argsort()[::-1]
53.
54.         if keep_top_k:
55.             order = order[:keep_top_k]
56.
57.         keep = []
58.
59.         while order.size > 0:
60.             i = order[0]
61.             keep.append(i)
62.
63.             xx1 = np.maximum(x1[i], x1[order[1:]])
64.             yy1 = np.maximum(y1[i], y1[order[1:]])
65.             xx2 = np.minimum(x2[i], x2[order[1:]])

```

```
66.         yy2 = np.minimum(y2[i], y2[order[1:]])
67.
68.         w = np.maximum(0.0, xx2 - xx1 + b)
69.         h = np.maximum(0.0, yy2 - yy1 + b)
70.         intersection = w * h
71.
72.         cw = np.maximum(x2[i], x2[order[1:]]) - np.mini
mum(x1[i], x1[order[1:]])
73.         ch = np.maximum(y2[i], y2[order[1:]]) - np.mini
mum(y1[i], y1[order[1:]])
74.         c_area = cw**2+ch**2+1e-16
75.         rh02 = ((x2[order[1:]]+x1[order[1:]])-(
(x2[i]+x1[i]))**2/4+((y2[order[1:]]+y1[order[1:]])-(
(y2[i]+y1[i]))**2/4
76.
77.         if use_min_area:
78.             base_area = np.minimum(areas[i], areas[orde
r[1:]])
79.         else:
80.             base_area = (areas[i] + areas[order[1:]] -
intersection)
81.
82.         overlap =
83.             np.divide( intersection
84.             n, base_area,
85.             out=np.zeros_like(intersection, dtype=float
),
86.             where=base_area != 0
87.             ) - pow(rh02/c_area,0.6)
88.         order = order[np.where(overlap <= thresh)[0] +
1] # pylint: disable=W0143
89.
90.         return keep
```

4.0 用 DIOU-NMS 检查 yolo-v4-tf 模型的精度

4.1. 下载和转换 yolo-v4-tf 模型

```
1. cd ~/Documents/  
2. python3 /opt/intel/openvino_2021/deployment_tools/tools/model_  
   downloader/downloader.py --name yolo-v4*  
3. python3 /opt/intel/openvino_2021/deployment_tools/tools/model_  
   downloader/converter.py --name yolo-v4*
```

4.2. 准备数据集和 yml 文件

4.2.1. 下载数据集

```
1. wget http://images.cocodataset.org/zips/val2017.zip  
2. wget http://images.cocodataset.org/annotations/annotations_trainval2017.zip  
3. unzip -  
   d annotations_trainval2017/ annotations_trainval2017.zip  
4. unzip -d annotations_trainval2017/ annotations_trainval2017.zip
```

4.2.2. 复制和编辑“accuracy-check.yml”

复制“accuracy-check.yml”

```
1. sudo mv /opt/intel/openvino_2021/deployment_tools/open_model_zoo/models/public/yolo-  
   v4-tf/accuracy-check.yml ~/Documents/
```

编辑“accuracy-check.yml”

1. 标注出第 2-60 行
2. 在第 65 行添加以下内容

```
1. - framework: dl sdk  
2.   tags:  
3.     - FP32  
4.   model: yolo-v4-tf.xml  
5.   weights: yolo-v4-tf.bin
```

3. 从以下位置开始编辑第 94 行

在 OpenVINO 精度检查器工具中添加 DIOU-NMS 指标，以获取 YOLO v4 的正确 mAP
白皮书

```
—type: nms  
—直到  
—type: diou-nms
```

4.3. 运行精度检查器工具，计算 yolo-v4-tf 的统计精度

如果上述设置和修改正确，请使用以下命令获取精度结果。

```
1. accuracy_check -c accuracy-check.yml -m public/yolo-v4-tf/FP32/ --  
definitions /opt/intel/openvino_2021/deployment_tools/open_model_zoo/tools/accuracy_checker/dataset_definitions.yml -s  
annotations_trainval2017/annotations/ -td CPU
```

mAP 结果：

yolo-v4-tf	DIOU-NMS	IoU-NMS
	71.23%	71.17%

结果证实 DIOU 的 mAP 比 IoU 更准确。尽管使用训练后优化工具包 (pot) 将模型转换为 INT8 时只有微小的差异，但是这些差异可能会影响转换后的 INT8 模型的性能。因此，实现面向 OpenVINO 工具™ (精度检查器和 pot) 的 DIOU-NMS 函数非常重要。

5.0

参考资料

参考文档	文档编号/位置
OpenVINO™	https://docs.openvinotoolkit.org/
OpenVINO Open Model Zoo	https://github.com/openvinotoolkit/open_model_zoo

§