

## MSCOCO物体检测评测系统的分析



哆来咪

Fortis Fortuna Aduvat

27 人赞同了该文章

检测指标通常用AP来衡量，现有的多个评测代码，比如COCO、VOC、Wider Face等等，AP的计算原理都是一样的，[github.com/huangh12/Obj...](https://github.com/huangh12/Obj...)即逼近PR曲线下的面积，因此从PR曲线到AP的计算大同小异。关键的差异在于PR曲线的获取上，各家的思路会有些差异，这个关键就反映在**det**与**gt**的**match策略**上。

在COCO中，gt还有一个crowd的属性，之前一直认为crowd就代表ignore，但是仔细看代码会发现，这个crowd属性和ignore其实是不同的，ignore是gt的另一个属性，

在计算AP的时候两者的处理方式不同，因此要格外注意。

各家评测系统的核心差异，主要就反映在det与gt的match策略上。COCO的匹配思想主要体现在下面一段代码中：

首先，det按照得分降序排列，gt按照是否ignore（0表示否1，表示是）升序排列，也就是把ignore为1的gt扔到后面去。

其匹配思想可以概括为：

1. 首先，得分优先原则，即得分大的det先去匹配gt；
2. 每次匹配的时候，每次匹配时的gt candidates是未被匹配的gt并上已匹配但是属性crowd为true的gt；
3. 从前到后遍历gt，从gt candidates中选择与当前det的iou最大的（需满足前提，即大于iou thresh）作为best match。
4. 匹配过程中若已经完成了一次常规匹配（reg match，也即det匹配到了一个非ignore的gt，注意这里只说了非ignore，并没有说非crowd），而且遍历到了gt的ignore=1部分，那么就趁早break，结束此det的match过程了。
5. 每次匹配过程结束（每次匹配过程就是一个det去找gt的过程），记录该det匹配到的gt的

赞同 27

12 条评论

分享

喜欢

收藏

申请转载

...

gt的

```

def evaluateImg(self, imgId, catId, aRng, maxDet):
    '''
    perform evaluation for single category and image
    :return: dict (single image results)
    '''
    p = self.params
    if p.useCats:
        gt = self._gts[imgId,catId]
        dt = self._dts[imgId,catId]
    else:
        gt = [_ for cId in p.catIds for _ in self._gts[imgId,cId]]
        dt = [_ for cId in p.catIds for _ in self._dts[imgId,cId]]
    if len(gt) == 0 and len(dt) ==0:
        return None

    for g in gt:
        if g['_ignore'] or (g['area']<aRng[0] or g['area']>aRng[1]):
            g['_ignore'] = 1
        else:
            g['_ignore'] = 0

    # sort dt highest score first, sort gt ignore last
    gtind = np.argsort([g['_ignore'] for g in gt], kind='mergesort')
    gt = [gt[i] for i in gtind]
    dtind = np.argsort([-d['score'] for d in dt], kind='mergesort')
    dt = [dt[i] for i in dtind[0:maxDet]]
    iscrowd = [int(o['iscrowd']) for o in gt]
    # load computed ious
    ious = self.ious[imgId, catId][:, gtind] if len(self.ious[imgId, catId]) >

    T = len(p.iouThrs)
    G = len(gt)
    D = len(dt)
    gtm = np.zeros((T,G))
    dtm = np.zeros((T,D))
    gtIg = np.array([g['_ignore'] for g in gt])
    dtIg = np.zeros((T,D))
    if not len(ious)==0:
        for tind, t in enumerate(p.iouThrs):
            for dind, d in enumerate(dt):
                # information about best match so far (m=-1 -> unmatched)
                iou = min([t,1-1e-10])
                m = -1
                for gind, g in enumerate(gt):
                    # if this gt already matched, and not a crowd, continue
                    if gtm[tind,gind]>0 and not iscrowd[gind]:
                        continue
                    # if dt matched to reg gt, and on ignore gt, stop
                    if m>-1 and gtIg[m]==0 and gtIg[gind]==1:
                        break
                    # continue to next gt unless better match made
                    if ious[dind,gind] < iou:
                        continue
                    # if match successful and best so far, store appropriately
                    iou=ious[dind,gind]
                    m=gind
                # if match made store id of match for both dt and gt
                if m ==-1:
                    continue
                dtIg[tind,dind] = gtIg[m]
                dtm[tind,dind] = gt[m]['id']
                gtm[tind,m] = d['id']

    # set unmatched detections outside of area range to ignore

```

rape(

赞同 27



12 条评论

分享

喜欢

收藏

申请转载



```
# store results for given image and category
return {
    'image_id':    imgId,
    'category_id': catId,
    'aRng':        aRng,
    'maxDet':      maxDet,
    'dtIds':       [d['id'] for d in dt],
    'gtIds':       [g['id'] for g in gt],
    'dtMatches':   dtm,
    'gtMatches':   gtm,
    'dtScores':    [d['score'] for d in dt],
    'gtIgnore':    gtIg,
    'dtIgnore':    dtIg,
}
```

COCO的匹配过程可以说是很细致了。从上面的匹配过程可以发现，评测时绝对不允许多个det匹配到同一个crowd为False的gt上（代码如下），但是却允许多个det匹配到crowd为True的gt上。

```
# if this gt already matched, and not a crowd, continue
if gtm[tind,gind]>0 and not iscrowd[gind]:
    continue
```

之所以对crowd为True的gt网开一面，看一下MSCOCO的原始论文中关于crowd的定义就明白了。

简言之，就是crowd的物体确实是有堆密集的东西在一起（比如一卡车香蕉），标注比较困难，所以就用crowd来减轻负担了。

For images containing 10 object instances or fewer of a given category, every instance was individually segmented (note that in some images up to 15 instances were segmented). Occasionally the number of instances is drastically higher; for example, consider a dense crowd of people or a truckload of bananas. In such cases, many instances of the same category may be tightly grouped together and distinguishing individual instances is difficult. After 10-15 instances of a category were segmented in an image, the remaining instances were marked as “crowds” using a single (possibly multi-part) segment. For the purpose of evaluation, areas marked as crowds will be ignored and not affect a detector’s score. Details are given in the appendix.

在cocoEval.py中set ignore flag的时候，ignore flag完全由crowd属性决定。

```
# set ignore flag
for gt in gts:
    gt['ignore'] = gt['ignore'] if 'ignore' in gt else 0
    gt['ignore'] = 'iscrowd' in gt and gt['iscrowd']
```

而ignore为True的gt样本在最后计算AP的时候没有起作用，所以crowd为True的gt确实没有影响到最后的AP。

COCO的评测代码为GT设计的crowd属性非常合理，考虑很细致，令人佩服。

继续回到刚才的匹配过程的代码，刚才说到了匹配时允许多个det匹配到crowd为True的gt上。但是，对于ignore为True的gt，却并没有说是否允许多个det匹配上去。

而这种设计是合理的！

因为，ignore属性在本质上其用途是为了在计算AP的时候不考虑这些gt，而crowd为True的gt是因为密集物体聚集（所以允许多个det匹配上去）。

赞同 27

12 条评论

分享

喜欢

收藏

申请转载

...

那么

此前，为了去掉crowd为True的gt对AP的影响，所以crowd为True的gt都被设成了ignore为True。

crowd属性与ignore属性，其含义与用途，其实已经反映在他们的名字上了。

那什么时候ignore的gt才允许多个det匹配上去呢？这个，完全取决于这个gt是否crowd为True，与ignore的取值无关。

例如，对于一个ignore为True，crowd为False的gt，匹配过程最多只允许一个det匹配上去，0个或者1个det匹配都不影响最终的AP，

但是如果有大量的det都检测的该gt，那么AP上就会有punishment，因为这些多余的det只能匹配到其他gt上，相当于false positive了。

而对于一个ignore为True，crowd为True的gt，匹配过程允许无数个det匹配上去，而且不影响最终的AP计算。

因此，需要摒弃之前所认为的crowd等同于ignore的思维旧势。

**crowd是region级别的，而ignore是instance级别的，一个严谨的检测标注规范应该独立标注二者，crowd用来表示密集聚集的一个区域，用ignore来忽略类标不明的一个instance个体。**

但是，在目前的实际业务中，只有ignore属性的标注，而且ignore属性事实上在业务中被当作crowd在用。

VOC和Wider Face也没有设置crowd属性，在计算AP的时候，直接把ignore当作crowd在用。而且匹配过程和COCO也不一样。

VOC以及Wider Face的匹配都是把det匹配到iou最大的一个gt上，这个匹配过程简单得就好像没有匹配过程一样。

发布于 2020-03-04 01:25

深度学习 (Deep Learning) 目标检测 物体检测

## 文章被以下专栏收录



**Detector**

千里之行，始于足下。

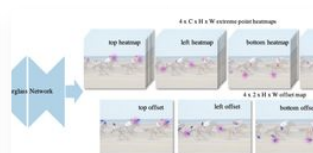
## 推荐阅读



### 目标检测之YOLOv2/3

lwzb

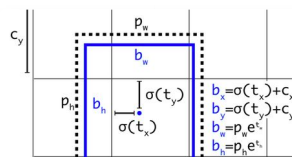
发表于Data ...



### 基于点检测的物体检测方法 (二)：ExtremeNet

autoc...

发表于计算机视觉...



### 目标检测：YOLOv2

Mr.AM...

发表于机器学习学...

### 目标检测合集 | 6 YOLOv2解析

YOLOv2/YOLO9000, Joseph Redmon在2016.12发表, YOLOv2引入了Anchor Box, 并用了很多改进; YOLO9000是在YOLOv2基础上提出的联合训练方法, 可以预测超过9000个不同目标类别。...

Mengc...

发表于目标检测合...

## 12 条评论

切换为时间排序

写下你的评论...



赞同 27

12 条评论

分享

喜欢

收藏

申请转载

...

11

不知理解的是否正确，Ignore与crowd的设计初衷就不同，一个是用于忽略某个gt实体，一个是针对密集的区域，但在目前的COCO评估中，gt的ignore指标还是按照crowd的属性而设置的，即crowd为1是ignore也为1. 通常将crowd当做了ignore来用。希望能得到解答，谢谢

👍 1

 哆来咪 (作者) 回复 知乎用户 2020-07-31

是的，crowd的gt都被设成了ignore，而ignore的gt未必都是crowd的。Crowd允许多个box匹配到该gt而不影响AP，ignore为true而crowd为false的gt，最多只允许一个box匹配上去，多余的视为false positive，会影响ap。不过，在coco里面，没有ignore为true而crowd为false的gt。


👍 4

 知乎用户 回复 哆来咪 (作者) 2020-07-31

谢谢解答

👍 赞

展开其他 1 条回复

 什么 01-18

👍

👍 赞

 种植苹果树 2021-07-13

觉得好厉害，最近也在尝试搞清楚目标检测指标的计算，被折磨死了，许多细节问题还是需要考究！

👍 赞

 逃避可耻但有用 2021-06-27

楼主文章写得很好，有个疑惑啊，现在coco数据集里可以定义"ignore"吗，我只看到了"iscrowd"

👍 赞

 哆来咪 (作者) 回复 逃避可耻但有用 2021-06-30

可以的，只是coco官方的标注数据没有使用ignore这个字段，但是如果你自己创建coco format的数据集时，可以利用这个ignore的字段的。

👍 赞

 逃避可耻但有用 回复 哆来咪 (作者) 2021-07-03

请问你的意思是自己定义ignore这个字段 然后在训练的时候自己去加逻辑ioa，防止梯度回传？

👍 赞

展开其他 1 条回复

 哆来咪 (作者) 2020-07-31

这个要自己实现了，通常大家都会选择丢弃crowd为true的样本。

👍 赞

 知乎用户 2020-07-31

这个是评估的时候忽略了crowd区域内的gt,还有一个疑惑，在训练时候能否也忽略crowd区域的梯度回传，具体实现的话COCO中有这方面的功能支持吗？

👍 赞