

# Readme

---

## 一、主要任务

### 二、算法思路

#### 2.1 核心方法

#### 2.2 数据处理

#### 2.3 数据增强

#### 2.4 数据分配

## 三、复现流程

#### 3.1 环境依赖

##### 3.1.1 docker依赖

##### 3.1.2 pip依赖

##### 3.1.3 mmdetection依赖

##### 3.1.4 Dockfile

#### 3.2 代码结构

#### 3.3 训练复现

##### 3.3.1 训练参数

##### 3.3.2 训练复现

#### 3.4 预测复现

##### 3.4.1 预测步骤

##### 3.4.2 预测复现

## 一、主要任务

比赛采用的数据为大量伪造的证书文档类图像。任务是通过提供的训练集学习出有效的检测算法，对测试集的伪造图像进行篡改定位。

## 二、算法思路

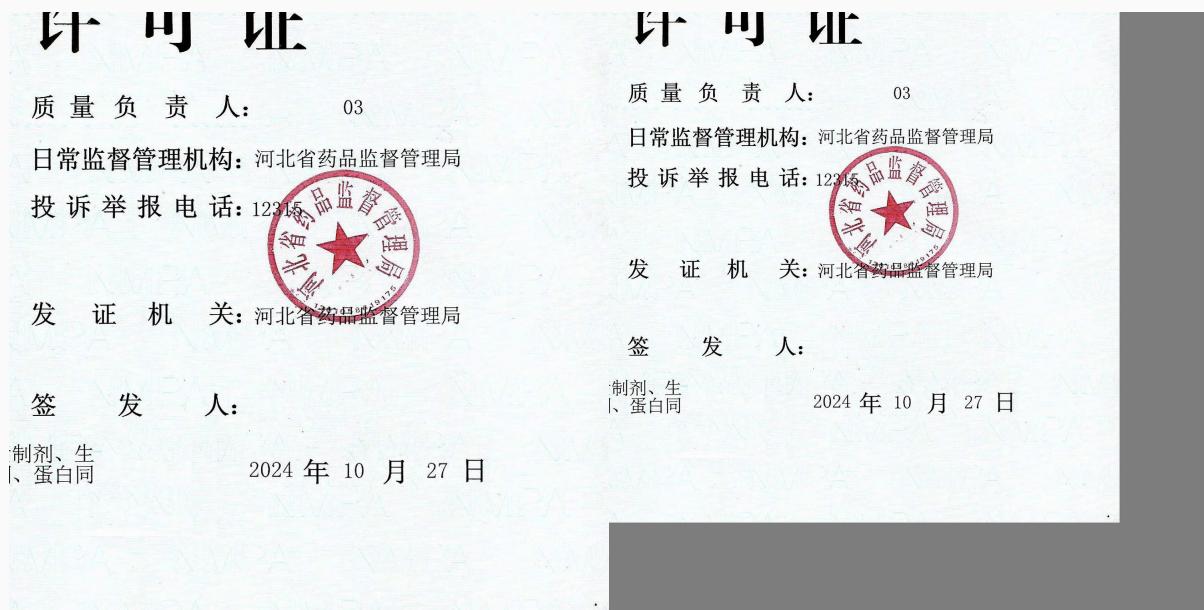
### 2.1 核心方法

转换为目标检测问题，将篡改区域定义目标类别**class-cheat**

## 2.2 数据处理

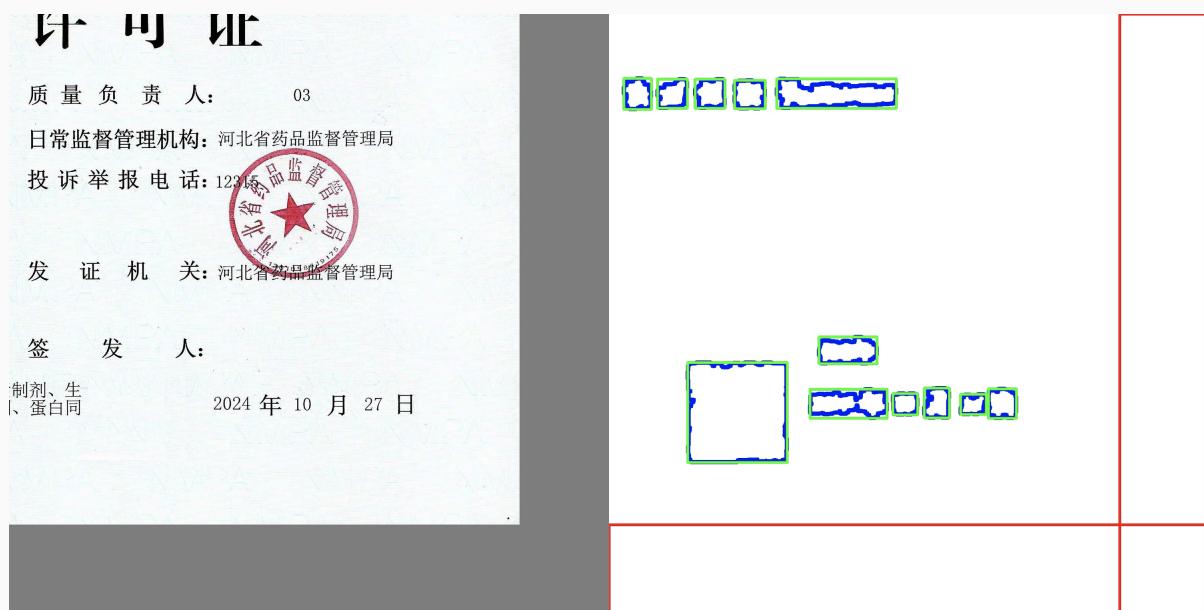
原始图像尺寸参差不齐，统一使用颜色125填充至1550\*1550。

举例：



颜色125部分定义为第二类别class-pmiss

对mask进行解析，如下红色为class-pmiss，蓝绿色为class-cheat。



据此构建COCO数据集，用于mmdetection目标检测。

## 2.3 数据增强

原有训练数据集较少，且部分图像来自同一张图像，为避免过拟合。使用了以下方法以数据增强。

- 原始

# 计 可 证

质量负责人: 03

日常监督管理机构: 河北省药品监督管理局

投诉举报电话: 12331

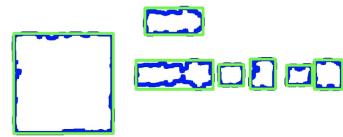


发证机关: 河北省药品监督管理局

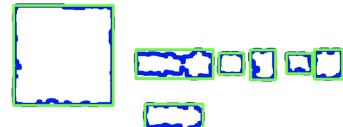
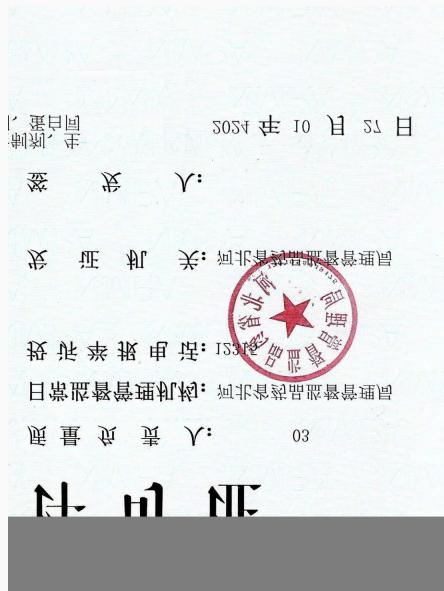
签发人:

制剂、生  
蛋白同

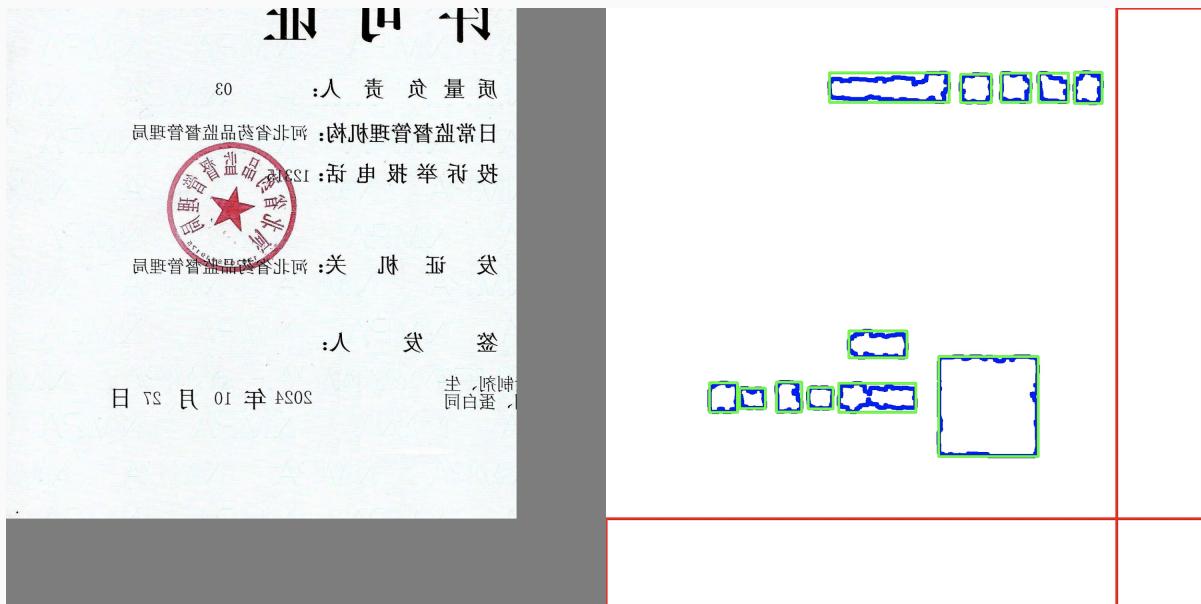
2024年10月27日



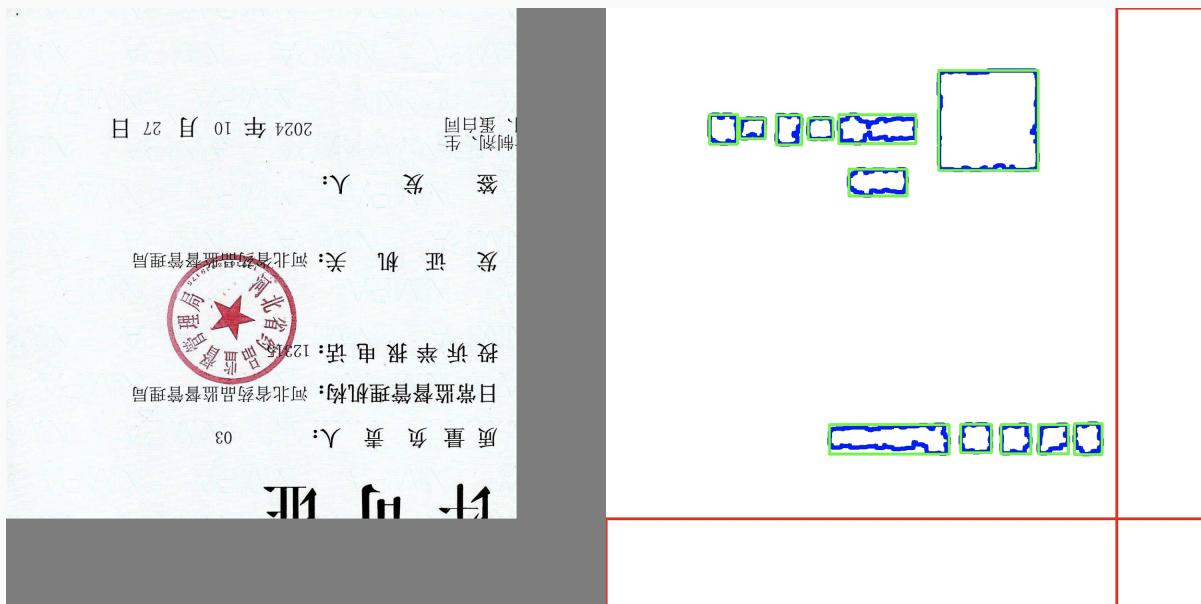
- cv旋转0



- cv旋转1



- cv旋转-1



- M2 (随机生成1/2边长正方形作为遮挡, 遮挡部分定义为class-pmiss)

# 计 可 证

质量负 责人: 03

日常监督管理

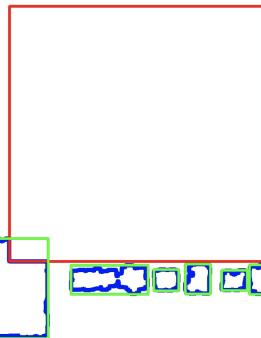
投诉举报

发 证 机

签 发

制剂、生  
蛋白同

2024 年 10 月 27 日



- MF (随机生成1/16边长正方形作为遮挡, 遮挡部分定义为class-pmiss)

# 计 可 证

质量负 责人: 03

日常监督管理机构: 河北省药品监督管理局



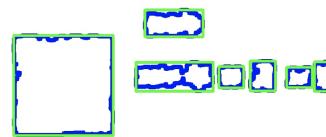
投诉举报电 话: 12331

发 证 机 关: 河北省药品监督管理局

签 发 人:

制剂、生  
蛋白同

2024 年 10 月 27 日



- blur (调整蓝色色调, 肉眼不可见)

# 计 可 证

质量负 责人: 03

日常监督管理机构: 河北省药品监督管理局

投诉举报电 话: 12331

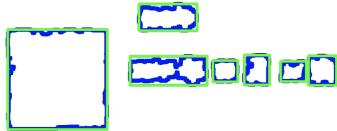


发 证 机 关: 河北省药品监督管理局

签 发 人:

制剂、生  
蛋白同

2024 年 10 月 27 日



- chicken (逛超市看到鸡块想到的鸡块法, 也就是随机删减)

# 计 可 证

质量负 责人: 03

日常监督管理机构: 河北省药品

投诉举报电 话: 12331



发 证 机 关: 河北省药品

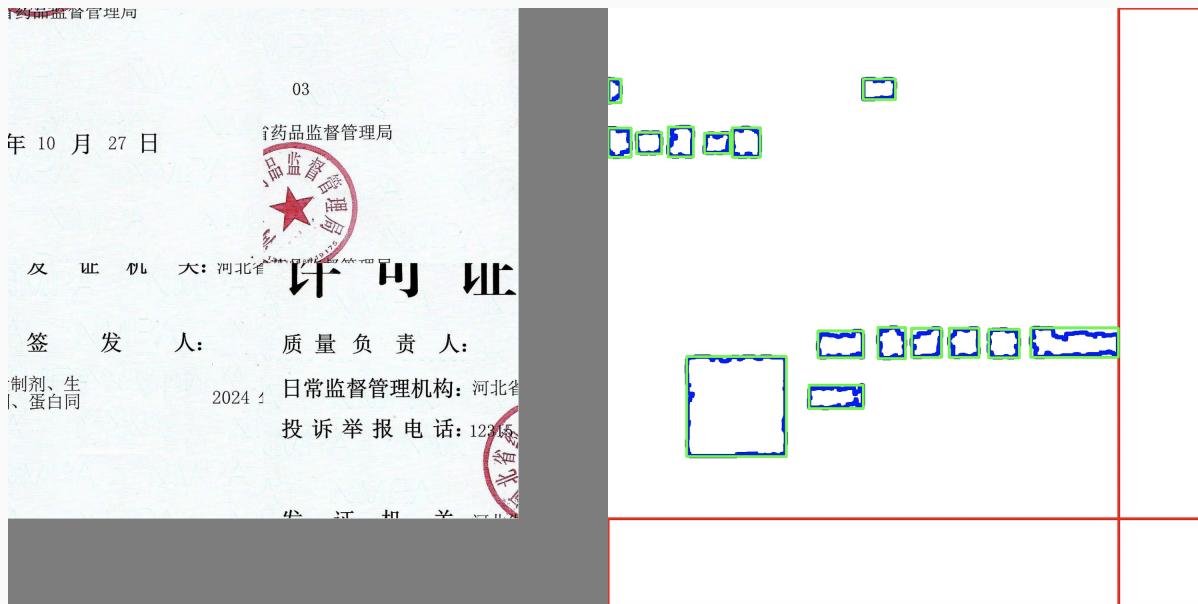
签 发 人:

制剂、生  
蛋白同

2024 年



- jelly (知名设计师jely提出的1/4随机拼凑)



时间有限，未逐一验证是否都有效，但本方案最终使用了除blur外全部。

详见 clean\_data\_code.py

## 2.4 数据分配

```
1 _n = random.choice(list("0000000012"))
```

随机分配，训练：测试：验证 = 8：1：1。

详见 clean\_data\_code.py

## 三、复现流程

### 3.1 环境依赖

#### 3.1.1 docker依赖

```
1 FROM registry.cn-shanghai.aliyuncs.com/tcc-public/pytorch:1.1.0-cuda10.0-py3
```

- 核心参数: cuda10.0
- 其他参数同此docker镜像

#### 3.1.2 pip依赖

```
1 ##
2 RUN apt-get update
3 RUN apt-get install libglib2.0-dev libsm6 libxrender1 libxext-dev
    -y
4
5 ## 在构建镜像时安装依赖包
6 RUN pip install -i https://mirrors.aliyun.com/pypi/simple -r requirements/build.txt
7 RUN pip install -i https://mirrors.aliyun.com/pypi/simple -r requirements/runtime.txt
```

- requirements/build.txt

pip==20.2.4

cython

pillow>=6.2.0

- requirements/runtime.txt

torch==1.3.1

torchvision==0.4.2

pandas

tqdm

opencv-python==4.2.0.34

pycocotools

### 3.1.3 mmdetection依赖

```
1 WORKDIR code/train/env/
2 RUN pip install mmcv_full-2.3.0+torch1.3.0+cu100-cp36-cp36m-manylinux1_x86_64.whl
3 WORKDIR mmdetection-master
4 RUN pip install -r requirements/build.txt
5 WORKDIR ../cocoapi-master/pycocotools/
6 RUN python setup.py build
7 RUN python setup.py install --user
8 WORKDIR ../../mmdetection-master
9 RUN pip install --no-cache-dir -e . --user
```

核心版本为mmcv\_full-2.3.0+torch1.3.0+cu100-cp36-cp36m-manylinux1\_x86\_64.whl

### 3.1.4 Dockfile

完整Dockfile如下：

```
1 # Base Images
2 ## 从天池基础镜像构建
3 FROM registry.cn-shanghai.aliyuncs.com/tcc-public/pytorch:1.1.0-c
  uda10.0-py3
4
5 ## 把当前文件夹里的文件构建到镜像的根目录下
6 ADD . /
7
8 ## 指定默认工作目录为根目录（需要把run.sh和生成的结果文件都放在该文件夹下，提交
  后才能运行）
9 WORKDIR /
10
11 ##
12 RUN apt-get update
13 RUN apt-get install libglib2.0-dev libsm6 libxrender1 libxext-dev
  -y
14
15 ## 在构建镜像时安装依赖包
16 RUN pip install -i https://mirrors.aliyun.com/pypi/simple -r requirements/build.txt
17 RUN pip install -i https://mirrors.aliyun.com/pypi/simple -r requirements/runtime.txt
18
19 ##
20 WORKDIR code/train/env/
21 RUN pip install mmcv_full-2.3.0+torch1.3.0+cu100-cp36-cp36m-manylinux1_x86_64.whl
22 WORKDIR mmdetection-master
23 RUN pip install -r requirements/build.txt
24 WORKDIR ../cocoapi-master/pycocotools/
25 RUN python setup.py build
26 RUN python setup.py install --user
27 WORKDIR ../../mmdetection-master
28 RUN pip install --no-cache-dir -e . --user
29
```

```
30 WORKDIR /
31 ## 镜像启动后统一执行 sh run.sh
32 CMD ["sh", "run.sh"]
```

## 3.2 代码结构

```
1 project
2     |--README.md          # 解决方案及算法介绍文件, 必选
3     |--requirements.txt    # Python环境依赖, 必选
4     |--s2_data
5         |--data
6             |--test
7                 |--1.jpg
8                 |--2.jpg
9                 ...
10                |--1500.jpg
11                |--train
12                    |--1.jpg
13                    |--2.jpg
14                    ...
15                    |--1500.jpg
16                |--train_mask
17                    |--1.png
18                    |--2.png
19                    ...
20                    |--1500.png
21            |--user_data
22                |--model_data      # 模型文件夹示例, 可自行组织
23                    |--model.pth
24                |--tmp_data        # 临时存储文件夹示例, 可自行组织
25                    |--tmp.dat
26                |-- 计算的中间文件
27                |-- 临时存储目录
28            |--prediction_result
29                |-- images
30                    |-- 1.png
31                    |-- 2.png
32                    ...
33                    |--1500.png
```

```
34     |--code  
35         |--train          # 训练代码文件夹示例，可自行组织  
36         |--test           # 预测代码文件夹示例，可自行组织  
37         |--run.sh         # 预测执行脚本，必选  
38         |--train.sh       # 训练执行脚本，必选  
39         |--clean_data_code.cpp(.py, .java) # 数据清洗代码，可自行取舍
```

同此结构。

## 3.3 训练复现

### 3.3.1 训练参数

训练模型为FasterRCNN (resnet50-19c8e357.pth, faster\_rcnn\_r50\_fpn\_1x\_coco)

获取官方文件后，我们做了部分修改，完整如下：

```
1 model = dict(  
2     type='FasterRCNN',  
3     pretrained='torchvision://resnet50',  
4     backbone=dict(  
5         type='ResNet',  
6         depth=50,  
7         num_stages=4,  
8         out_indices=(0, 1, 2, 3),  
9         frozen_stages=1,  
10        norm_cfg=dict(type='BN', requires_grad=True),  
11        norm_eval=True,  
12        style='pytorch'),  
13     neck=dict(  
14         type='FPN',  
15         in_channels=[256, 512, 1024, 2048],  
16         out_channels=256,  
17         num_outs=5),  
18     rpn_head=dict(  
19         type='RPNHead',  
20         in_channels=256,  
21         feat_channels=256,  
22         anchor_generator=dict(  
23             type='AnchorGenerator',  
24             scales=[8],  
25             ratios=[0.5, 1.0, 2.0],
```

```

26             strides=[4, 8, 16, 32, 64]),
27             bbox_coder=dict(
28                 type='DeltaXYWHBoxCoder',
29                 target_means=[0.0, 0.0, 0.0, 0.0],
30                 target_stds=[1.0, 1.0, 1.0, 1.0]),
31             loss_cls=dict(
32                 type='CrossEntropyLoss', use_sigmoid=True, loss_weight=
ht=1.0),
33                 loss_bbox=dict(type='L1Loss', loss_weight=1.0)),
34             roi_head=dict(
35                 type='StandardRoIHead',
36                 bbox_roi_extractor=dict(
37                     type='SingleRoIExtractor',
38                     roi_layer=dict(type='RoIAlign', output_size=7, sampl
ing_ratio=0),
39                     out_channels=256,
40                     featmap_strides=[4, 8, 16, 32])),
41             bbox_head=dict(
42                 type='Shared2FCBBoxHead',
43                 in_channels=256,
44                 fc_out_channels=1024,
45                 roi_feat_size=7,
46                 num_classes=2,
47                 bbox_coder=dict(
48                     type='DeltaXYWHBoxCoder',
49                     target_means=[0.0, 0.0, 0.0, 0.0],
50                     target_stds=[0.1, 0.1, 0.2, 0.2]),
51                 reg_class_agnostic=False,
52                 loss_cls=dict(
53                     type='CrossEntropyLoss', use_sigmoid=False, loss
_weight=1.0),
54                     loss_bbox=dict(type='L1Loss', loss_weight=1.0))))
55 train_cfg = dict(
56     rpn=dict(
57         assigner=dict(
58             type='MaxIoUAssigner',
59             pos_iou_thr=0.7,
60             neg_iou_thr=0.3,
61             min_pos_iou=0.3,
62             match_low_quality=True,

```

```

63             ignore_iof_thr=-1),
64             sampler=dict(
65                 type='RandomSampler',
66                 num=256,
67                 pos_fraction=0.5,
68                 neg_pos_ub=-1,
69                 add_gt_as_proposals=False),
70                 allowed_border=-1,
71                 pos_weight=-1,
72                 debug=False),
73             rpn_proposal=dict(
74                 nms_across_levels=False,
75                 nms_pre=2000,
76                 nms_post=1000,
77                 max_num=1000,
78                 nms_thr=0.7,
79                 min_bbox_size=0),
80             rcnn=dict(
81                 assigner=dict(
82                     type='MaxIoUAssigner',
83                     pos_iou_thr=0.5,
84                     neg_iou_thr=0.5,
85                     min_pos_iou=0.5,
86                     match_low_quality=False,
87                     ignore_iof_thr=-1),
88                 sampler=dict(
89                     type='RandomSampler',
90                     num=512,
91                     pos_fraction=0.25,
92                     neg_pos_ub=-1,
93                     add_gt_as_proposals=True),
94                     pos_weight=-1,
95                     debug=False))
96 test_cfg = dict(
97     rpn=dict(
98         nms_across_levels=False,
99         nms_pre=1000,
100        nms_post=1000,
101       max_num=1000,
102       nms_thr=0.7,

```

```

103         min_bbox_size=0),
104     rcnn=dict(
105         score_thr=0.05,
106         nms=dict(type='nms', iou_threshold=0.5),
107         max_per_img=100))
108 dataset_type = 'CocoDataset'
109 data_root = 'data/coco/'
110 img_norm_cfg = dict(
111     mean=[123.675, 116.28, 103.53], std=[58.395, 57.12, 57.375],
112     to_rgb=True)
113 train_pipeline = [
114     dict(type='LoadImageFromFile'),
115     dict(type='LoadAnnotations', with_bbox=True),
116     dict(type='Resize', img_scale=(1550, 1550), keep_ratio=True)
117     ,
118     dict(type='RandomFlip', flip_ratio=0.5),
119     dict(
120         type='Normalize',
121         mean=[123.675, 116.28, 103.53],
122         std=[58.395, 57.12, 57.375],
123         to_rgb=True),
124     dict(type='Pad', size_divisor=32),
125     dict(type='DefaultFormatBundle'),
126     dict(type='Collect', keys=['img', 'gt_bboxes', 'gt_labels'])
127 ]
128 test_pipeline = [
129     dict(type='LoadImageFromFile'),
130     dict(
131         type='MultiScaleFlipAug',
132         img_scale=(1550, 1550),
133         flip=False,
134         transforms=[
135             dict(type='Resize', keep_ratio=True),
136             dict(type='RandomFlip'),
137             dict(
138                 type='Normalize',
139                 mean=[123.675, 116.28, 103.53],
140                 std=[58.395, 57.12, 57.375],
141                 to_rgb=True),
142             dict(type='Pad', size_divisor=32),

```

```

141             dict(type='ImageToTensor', keys=['img']),
142             dict(type='Collect', keys=['img']))
143         ])
144     ]
145 data = dict(
146     samples_per_gpu=2,
147     workers_per_gpu=2,
148     train=dict(
149         type='CocoDataset',
150         ann_file='code/train/env/mmdetection-master/data/coco/annotations/annotations_train.json',
151         img_prefix='code/train/env/mmdetection-master/data/coco/img/',
152         pipeline=[
153             dict(type='LoadImageFromFile'),
154             dict(type='LoadAnnotations', with_bbox=True),
155             dict(type='Resize', img_scale=(1550, 1550), keep_ratio=True),
156             dict(type='RandomFlip', flip_ratio=0.5),
157             dict(
158                 type='Normalize',
159                 mean=[123.675, 116.28, 103.53],
160                 std=[58.395, 57.12, 57.375],
161                 to_rgb=True),
162             dict(type='Pad', size_divisor=32),
163             dict(type='DefaultFormatBundle'),
164             dict(type='Collect', keys=['img', 'gt_bboxes', 'gt_labels']))
165         ],
166     val=dict(
167         type='CocoDataset',
168         ann_file='code/train/env/mmdetection-master/data/coco/annotations/annotations_test.json',
169         img_prefix='code/train/env/mmdetection-master/data/coco/img/',
170         pipeline=[
171             dict(type='LoadImageFromFile'),
172             dict(
173                 type='MultiScaleFlipAug',
174                 img_scale=(1550, 1550),

```

```

175         flip=False,
176         transforms=[
177             dict(type='Resize', keep_ratio=True),
178             dict(type='RandomFlip'),
179             dict(
180                 type='Normalize',
181                 mean=[123.675, 116.28, 103.53],
182                 std=[58.395, 57.12, 57.375],
183                 to_rgb=True),
184             dict(type='Pad', size_divisor=32),
185             dict(type='ImageToTensor', keys=['img']),
186             dict(type='Collect', keys=['img'])
187         ])
188     ],
189     test=dict(
190         type='CocoDataset',
191         ann_file='code/train/env/mmdetection-master/data/coco/annotations/annotations_vals.json',
192         img_prefix='code/train/env/mmdetection-master/data/coco/img/',
193         pipeline=[
194             dict(type='LoadImageFromFile'),
195             dict(
196                 type='MultiScaleFlipAug',
197                 img_scale=(1550, 1550),
198                 flip=False,
199                 transforms=[
200                     dict(type='Resize', keep_ratio=True),
201                     dict(type='RandomFlip'),
202                     dict(
203                         type='Normalize',
204                         mean=[123.675, 116.28, 103.53],
205                         std=[58.395, 57.12, 57.375],
206                         to_rgb=True),
207                     dict(type='Pad', size_divisor=32),
208                     dict(type='ImageToTensor', keys=['img']),
209                     dict(type='Collect', keys=['img'])
210                 ])
211             ]))
212 evaluation = dict(interval=1, metric='bbox')

```

```

213 optimizer = dict(type='SGD', lr=0.02, momentum=0.9, weight_decay
214     =0.0001)
215 optimizer_config = dict(grad_clip=None)
216 lr_config = dict(
217     policy='step',
218     warmup='linear',
219     warmup_iters=500,
220     warmup_ratio=0.001,
221     step=[8, 11])
222 total_epochs = 1
223 checkpoint_config = dict(interval=1)
224 log_config = dict(interval=50, hooks=[dict(type='TextLoggerHook'
225 )])
226 dist_params = dict(backend='nccl')
227 log_level = 'INFO'
228 load_from = None
229 resume_from = None
230 workflow = [('train', 2), ('val', 1)]
231 work_dir = 'code/train/env/mmdetection-master/work_dirs'
232 gpu_ids = range(0, 1)

```

可以换用其他更高版本。

完整训练代码见 `code/train.sh`, 一共分三步:

- (1) `clean_data_code`训练数据处理;
- (2) `mmdetection`初始化, 主要导入模型、创建路径、修改文件;
- (3) 开始训练;

```

1 # Train
2 # data
3 cd /
4 python code/clean_data_code.py train
5
6 # model
7 mmdetn=1
8 mmdet=code/train/env/mmdetection-master
9
10 mkdir -p /root/.cache/torch/checkpoints
11 cp user_data/model_data/resnet50-19c8e357.pth /root/.cache/torch/

```

```
checkpoints/resnet50-19c8e357.pth
12
13 mkdir -p $mmdet/data/coco
14 ln -s /user_data/tmp_data/work $mmdet/data/coco/annotations
15 ln -s /user_data/tmp_data/work $mmdet/work_dirs
16
17 cp code/train/coco$mmdetn.py $mmdet/mmdet/datasets/coco.py
18 cp code/train/class_names$mmdetn.py $mmdet/mmdet/core/evaluation/
    class_names.py
19
20 # 示例total_epochs = 1, 复现需修改
21 python $mmdet/tools/train.py code/train/faster_rcnn_r50_fpn_1x_co
    co.py --gpus 1 --work-dir $mmdet/work_dirs
```

其中，

- coco.py需修改：

```
1 @DATASETS.register_module()
2 class CocoDataset(CustomDataset):
3
4     # CHANGE BY IVAN
5     CLASSES = ('000', '001')
6     # CHANGE BY IVAN
```

- class\_names.py需修改：

```
1 # CHANGE BY IVAN
2 def coco_classes():
3     return [
4         '000',
5         '001'
6     ]
7 # CHANGE BY IVAN
```

### 3.3.2 训练复现

```
1 cd /
```

```
2 sh code/train.sh
```

但应修改`code/train/faster_rcnn_r50_fpn_1x_coco.py`内训练次数（现为1）

## 3.4 预测复现

### 3.4.1 预测步骤

(1) `clean_data_code`预测数据处理，主要修改尺寸；

```
1 # test
2 nnn = 0
3 test_ot = f"{ot}/temp3"
4 for i in tqdm(os.listdir(f"{dt}/test")):
5     if ".jpg" in i:
6         i = i.replace(".jpg", "")
7         img0 = np.random.randint(0, 1, size=(N,N,3), dtype=np.uint8)
8         img0[:, :] = P
9
10        imgb = cv.imread(f"{dt}/test/{i}.jpg")
11        i_y, i_x, _ = imgb.shape
12
13        img0[:i_y, :i_x] = imgb
14        cv.imwrite(f"{test_ot}/{i}.jpg", img0)
15
16        nnn += 1
17        if nnn > 9999:
18            break
```

(2) 导入模型-S499、S488、S477，这里的4XX值模型线上分数；

```
1 #
2 import os
3 import cv2 as cv
4 import numpy as np
5 import pandas as pd
6 from tqdm import tqdm
7 from mmdet/apis import init_detector, inference_detector
```

```

8
9 config_file = "code/train/faster_rcnn_r50_fpn_1x_coco.py"
10 checkpoint_file = "user_data/model_data/S499.pth"
11 model1 = init_detector(config_file, checkpoint_file, device="cuda:0")
12 print("M1", model1.CLASSES)
13
14 checkpoint_file = "user_data/model_data/S488.pth"
15 model2 = init_detector(config_file, checkpoint_file, device="cuda:0")
16 print("M2", model2.CLASSES)
17
18 checkpoint_file = "user_data/model_data/S477.pth"
19 model3 = init_detector(config_file, checkpoint_file, device="cuda:0")
20 print("M3", model3.CLASSES)

```

(3) 调用模型，分别获取预测结果；

```

1 #
2 N = 9999
3 p5 = "user_data/tmp_data/temp3/"
4 p6 = "s2_data/data/test/"
5 p7 = "prediction_result/images/"
6
7 dresult1, dresult2, dresult3 = {}, {}, {}
8
9 nnn = 0
10 for _img in tqdm(os.listdir(f"{p5}")):
11     # print(_img)
12     if ".jpg" in _img:
13         _img = _img.replace(".jpg", "").replace("_", "")
14
15         result = inference_detector(model1, f"{p5}_{_img}.jpg")
16         _result = [ij for i, j in enumerate(result) if i == 0 and
17         j.shape != (0,5) for ij in j]
18         dresult1[_img] = _result
19
20         result = inference_detector(model2, f"{p5}_{_img}.jpg")

```

```
20         _result = [ij for i, j in enumerate(result) if i == 0 and
21             j.shape != (0,5) for ij in j]
22         dresult2[_img] = _result
23
24         result = inference_detector(model3, f"p5_{_img}.jpg")
25         _result = [ij for i, j in enumerate(result) if i == 0 and
26             j.shape != (0,5) for ij in j]
27         dresult3[_img] = _result
28
29         nnn += 1
30         if nnn > N:
31             break
```

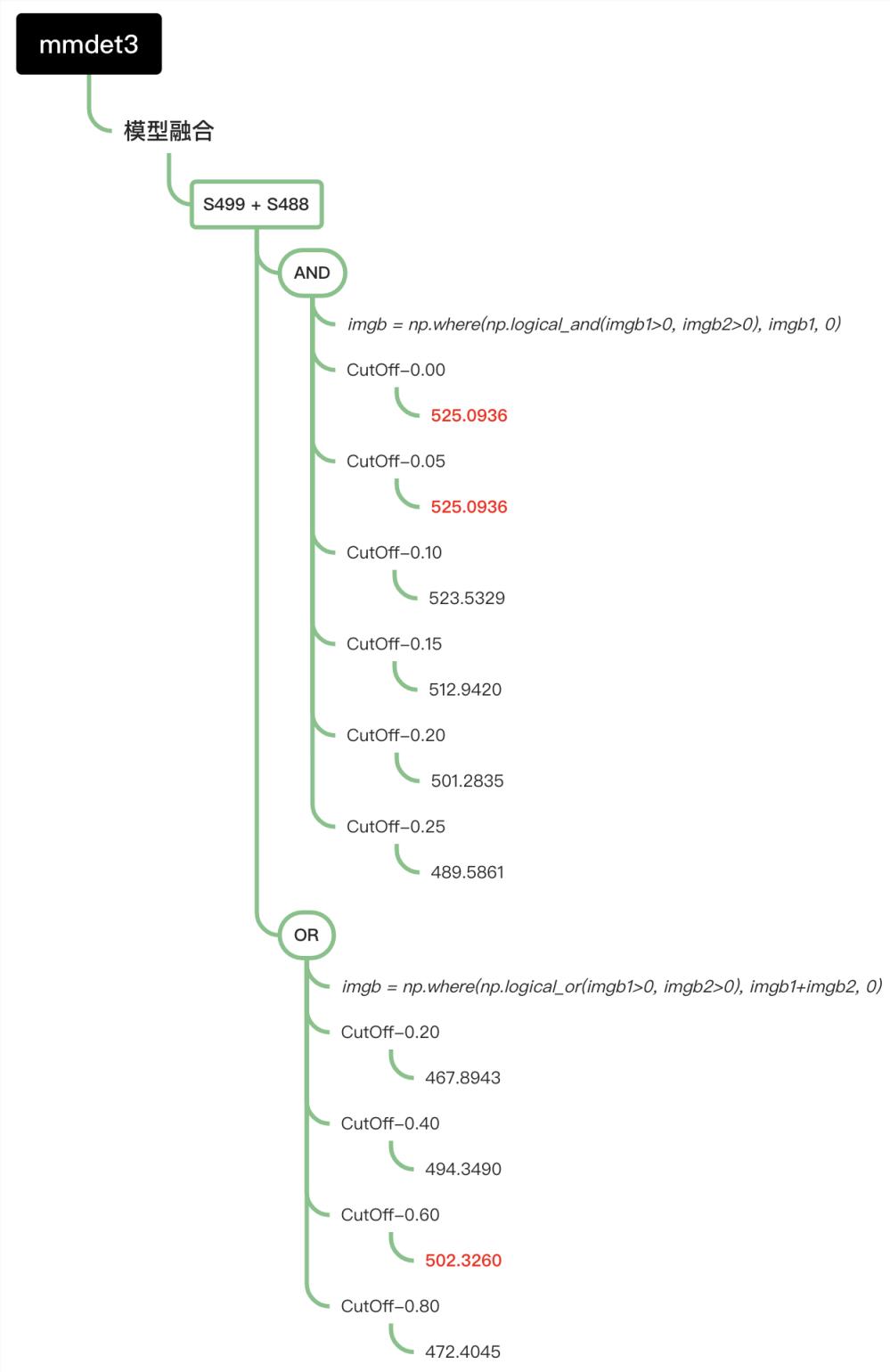
#### (4) 结果输出

这里我们在训练集上做了大量探索，引入模型融合概念。

调整方法有三：

- 预测概率阈值，小icut即放弃；
- 大小限制，预测框过大或过小即放弃；
- 不同合并方式，取三个模型预测mask的并集或交集或其他；

对不同阈值、不同合并方式，线上情况如下：



可提高50分左右。

最终选定预测概率阈值为0，限定预测框大小300–90000，并取三个模型预测mask交集（**最小化、最精确化**）。

```

1 #
2icut = 0.00

```

```

3 high1, high2 = 90000, 300 # 90000, 300
4 logs = "and"
5 r1, r2, r3 = [], [], []
6
7 nnn = 0
8 for _img in tqdm(os.listdir(f"{p5}")):
9     if ".jpg" in _img:
10         _img = _img.replace(".jpg", "").replace("_", "")
11
12
13     #
14     imgb1 = cv.imread(f"{p6}{_img}.jpg")
15     imgb1[:] = 0
16     for _iresult in dresult1[_img]:
17         ymin, ymax, xmin, xmax = int(_iresult[1]), int(_iresu
18             lt[3]), int(_iresult[0]), int(_iresult[2])
19         if _iresult[4] >= icut and (high1 >= (ymax-ymin)*(xma
20             x-xmin) >= high2):
21             r1.append((ymax-ymin)*(xmax-xmin))
22             imgb1[ymin:ymax, xmin:xmax] = bads
23
24     #
25     imgb2 = cv.imread(f"{p6}{_img}.jpg")
26     imgb2[:] = 0
27     for _iresult in dresult2[_img]:
28         ymin, ymax, xmin, xmax = int(_iresult[1]), int(_iresu
29             lt[3]), int(_iresult[0]), int(_iresult[2])
30         if _iresult[4] >= icut and (high1 >= (ymax-ymin)*(xma
31             x-xmin) >= high2):
32             r2.append((ymax-ymin)*(xmax-xmin))
33             imgb2[ymin:ymax, xmin:xmax] = bads
34
35     #
36     imgb3 = cv.imread(f"{p6}{_img}.jpg")
37     imgb3[:] = 0
38     for _iresult in dresult3[_img]:
39         iymin, ymax, xmin, xmax = int(_iresult[1]), int(_iresu
40             lt[3]), int(_iresult[0]), int(_iresult[2])
41         if _iresult[4] >= icut and (high1 >= (ymax-ymin)*(xma
42             x-xmin) >= high2):

```

```

37             r3.append((ymax-ymin)*(xmax-xmin))
38             imgb3[ymin:ymax, xmin:xmax] = bads
39
40
41         if logs == "and":
42             imgb = np.where(np.logical_and(imgb1>0, imgb2>0, imgb
43 >0), imgb1, 0)
44         if logs == "or":
45             imgb = np.where(np.logical_or(imgb1>0, imgb2>0, imgb3
46 >0), imgb1+imgb2+imgb3, 0)
47
48         cv.imwrite(f"p7}{_img}.png", imgb)
49
50         nnn += 1
51         if nnn > N:
52             break

```

### 3.4.1 预测复现

```

1 # Tests
2 # data
3 cd /
4 python code/clean_data_code.py test
5
6 # model
7 python code/test/result.py

```

```

1 cd /
2 sh code/run.sh

```