

图像（目标）检测入门级理论课程

李晨阳
达摩院-开放视觉智能实验室

ModelScope交流群



OpenVI对外交流钉钉群

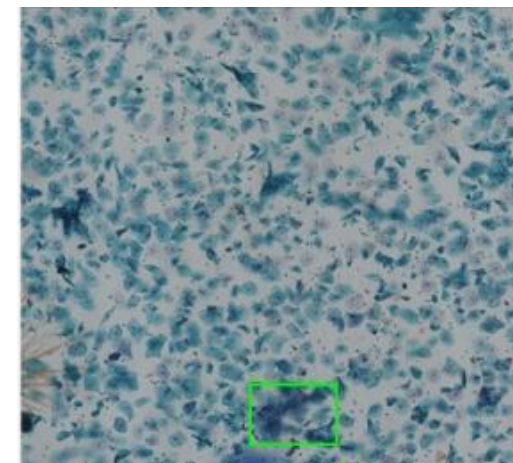




人体计数



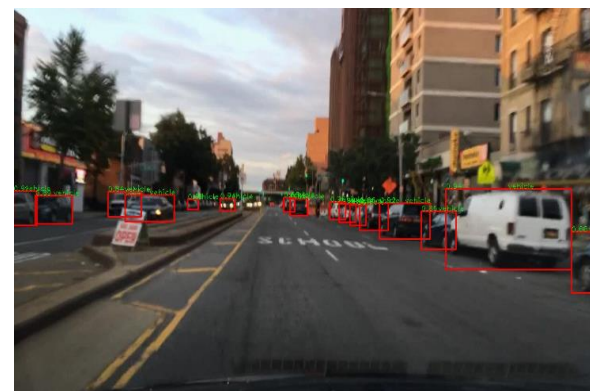
缺陷检测



病灶检测



智能座舱



自动驾驶

课程目标

01 掌握什么是目标检测

02 了解目标检测常用的数据集和评价指标

03 了解代表性的目标检测方法

Contents

目录

01 基础知识

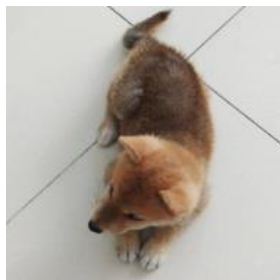
02 代表性方法

03 总结展望

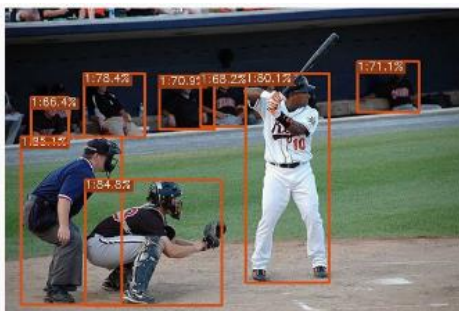
01 基础知识

目标检测的定义

图像分类：图像中主要包含一个物体，识别该物体的**类别**，并给出**分数**

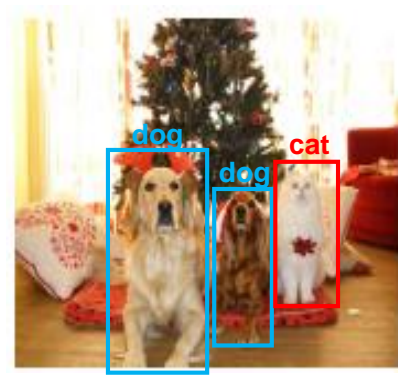


目标检测：对于有一个或多个目标的图片，检测出所有目标所处的**位置**，分类其**类别**，并给出**分数**

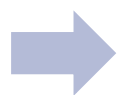
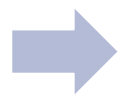


目标检测的输入输出格式

数据标注：使用标注工具，每个目标使用**矩形框**和**类别**表示，左上角坐标+右下角坐标 **(x1, y1, x2, y2, class)** 。



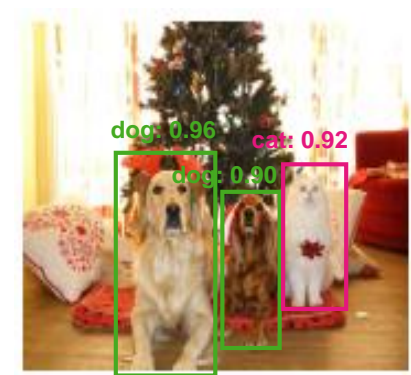
输入输出：输入为一张图像；输出为若干个物体的检测结果，包括：检测框、类别和分数，即： **(x1, y1, x2, y2, class, score)**



检测框
(x1, y1, x2, y2)
(x1, y1, x2, y2)
(x1, y1, x2, y2)

类别
dog
dog
cat

分数
0.96
0.90
0.92



目标检测常用数据集

PASCAL VOC:

- PASCAL全称是**P**attern **A**nalysis, **S**tatistical **M**odelling and **C**omputational **L**earning, VOC的全称是**V**isual **O**bject **C**lasses
- 从2005年到2012年, 每年都举办一场图像识别比赛。该数据集包含**20**类目标: person, bird, cat, cow, dog, horse, sheep, aeroplane, bicycle, boat, bus, car, motorbike, train, bottle, chair, dining table, potted plant, sofa, tv/monitor
- PASCAL VOC主要有 Object Classification, Object Detection, Object Segmentation, Human Layout, Action Classification 五类子任务



目标检测常用数据集

MS COCO:

- 全称是**MicroSoft Common Objects in COntext**, 主要从复杂的日常场景中截取
- 30W+图片, 250W+实例
- 提供**80**个目标类别

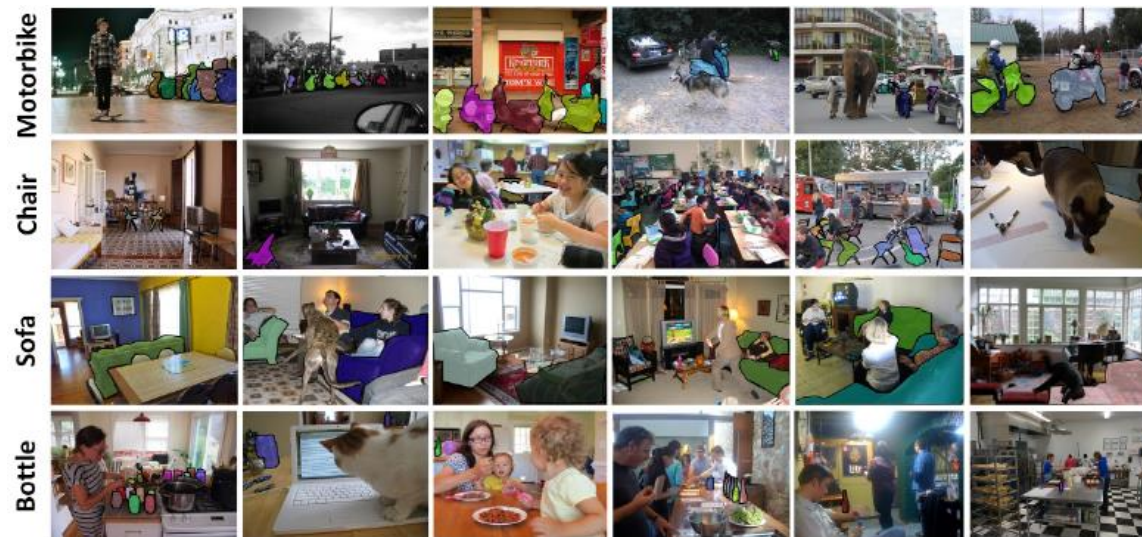
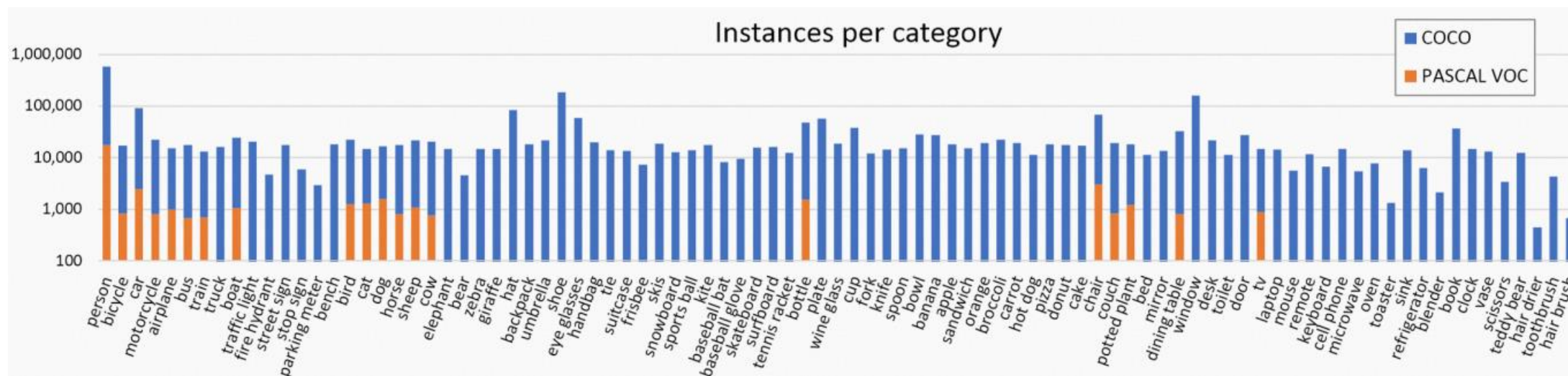


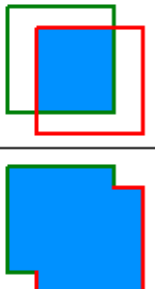
Fig. 6: Samples of annotated images in the MS COCO dataset.



目标检测重要操作

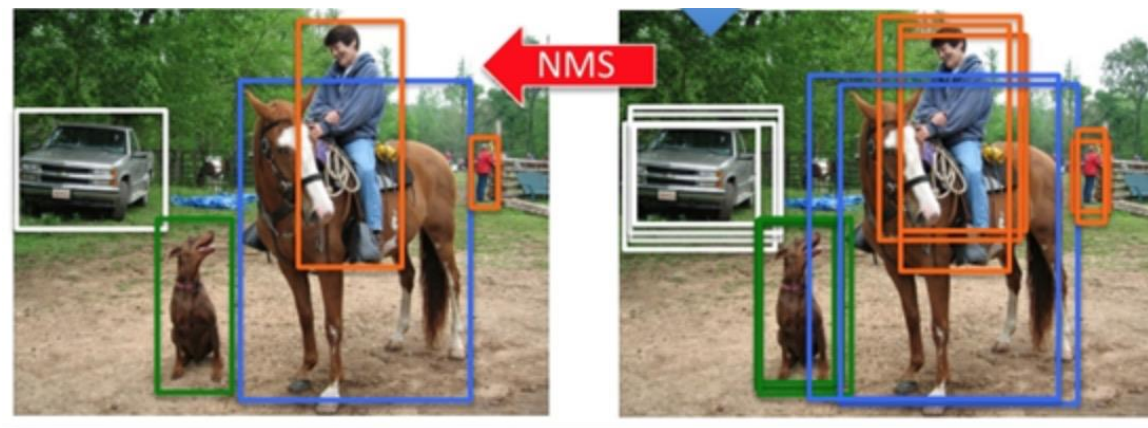
IoU: Intersection over Union

- 表示两个矩形框的重叠程度

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{area of overlap}}{\text{area of union}}$$


NMS: Non-Maximum Suppression

- 去掉多个重复的预测框
- 流程：
 - 设定一个IoU阈值
 - 逐类别分数排序
 - 计算IoU并进行抑制



以人体橙色的5个框为例:

x1,y1,x2,y2,0.91
x1,y1,x2,y2,0.80
x1,y1,x2,y2,0.79

x1,y1,x2,y2,0.85
x1,y1,x2,y2,0.81

目标检测评价指标

精度:

1. 准确率 (Precision)
2. 召回率 (Recall)
3. PR曲线
4. AP (Average Precision)
5. mAP (mean Average Precision)
6. ...

$$\text{准确率(Precision)} = \frac{\text{检测出的正确的框的数量}}{\text{检测出的所有框数量}}$$

$$\text{召回率(Recall)} = \frac{\text{检测出的正确的框的数量}}{\text{标注的所有框的数量}}$$

速度:

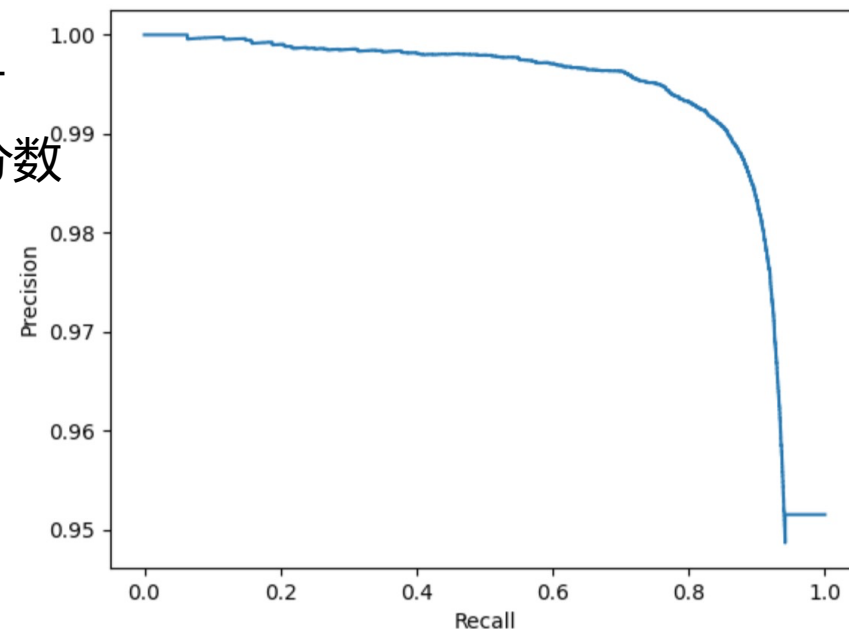
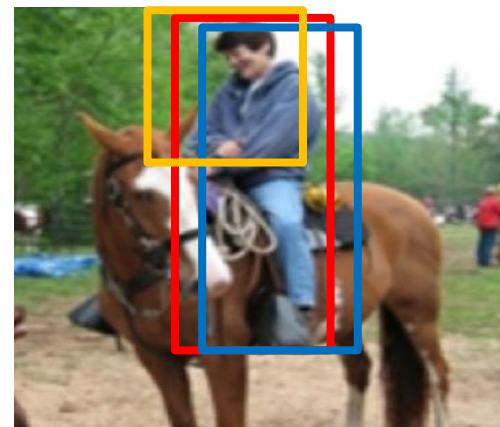
1. 运算量 (FLOPS)
2. 模型大小
3. 帧每秒 (FPS)
4. ...

PR曲线: 每一个置信度分数阈值对应一对 precision/recall 的值, 从 0 ~ 1 取一系列分数阈值就构成了一条曲线

AP: PR曲线与坐标轴构成区域的面积

mAP: 多个类别时所有类别的平均AP

红框: 标注框
蓝框: 预测框, 分数0.9
橙框: 预测框, 分数0.6



02 代表性方法

Object Detection on COCO minival

Leaderboard

Dataset

View

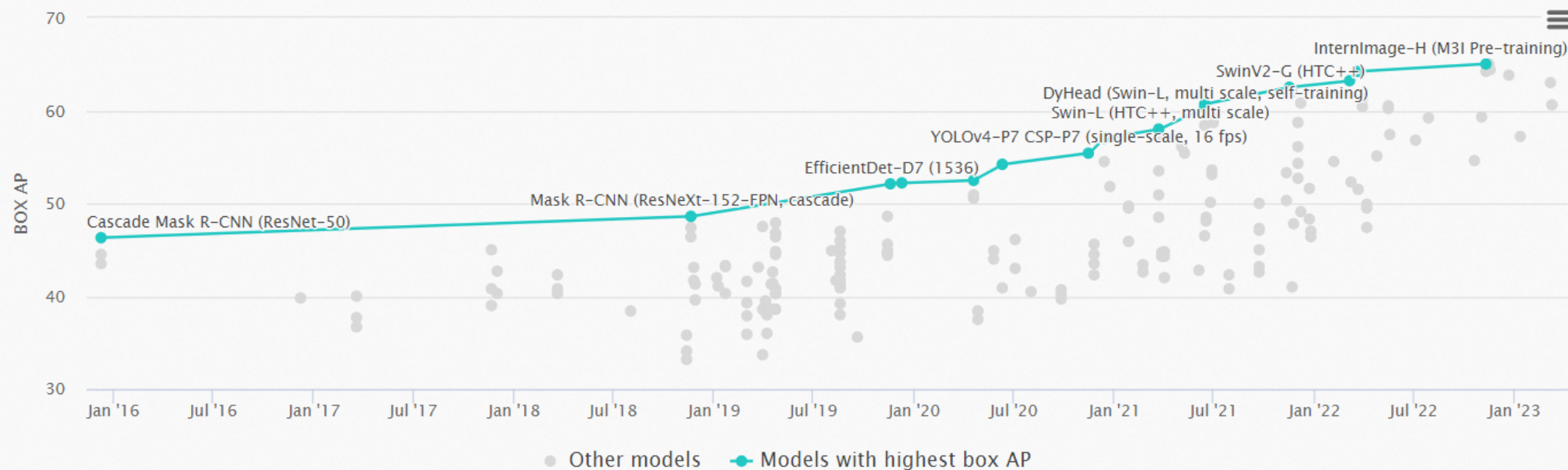
box AP

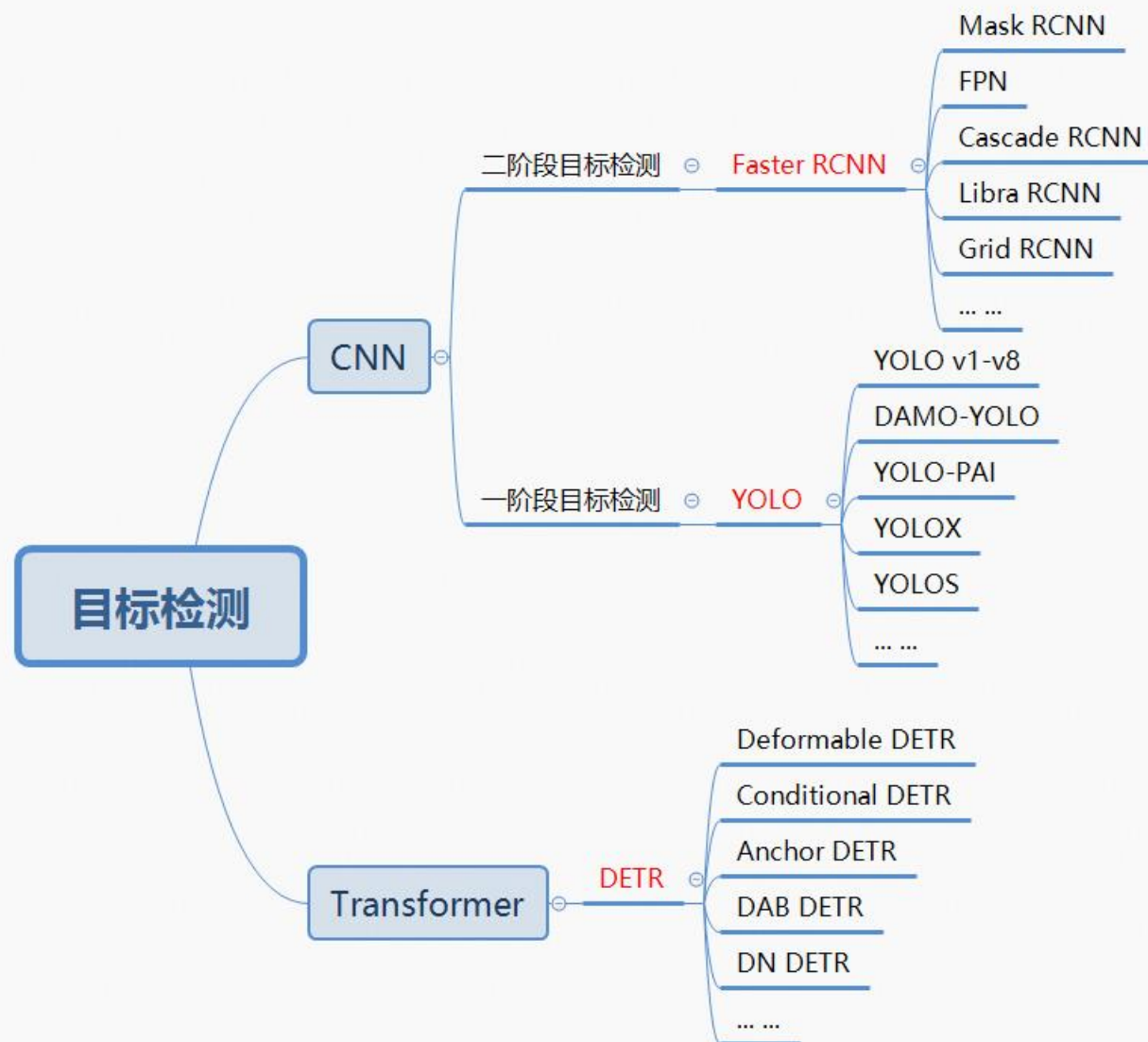
by

Date

for

All models





二阶段目标检测—Faster RCNN

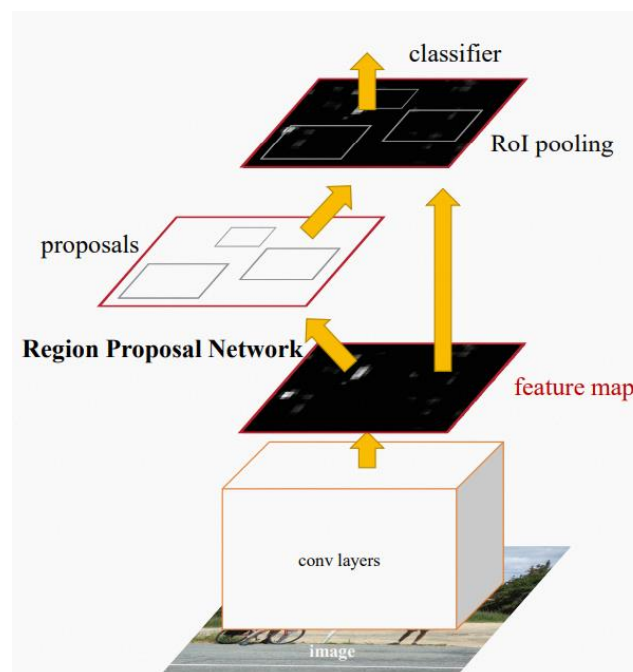
传统方法:

- 使用滑动窗口从原图裁剪图片进行分类和定位
- 不足:
 - 效率较低, 重复提取特征;
 - 精度较低, 对尺寸、光照等变化不鲁棒



Faster RCNN^[1]:

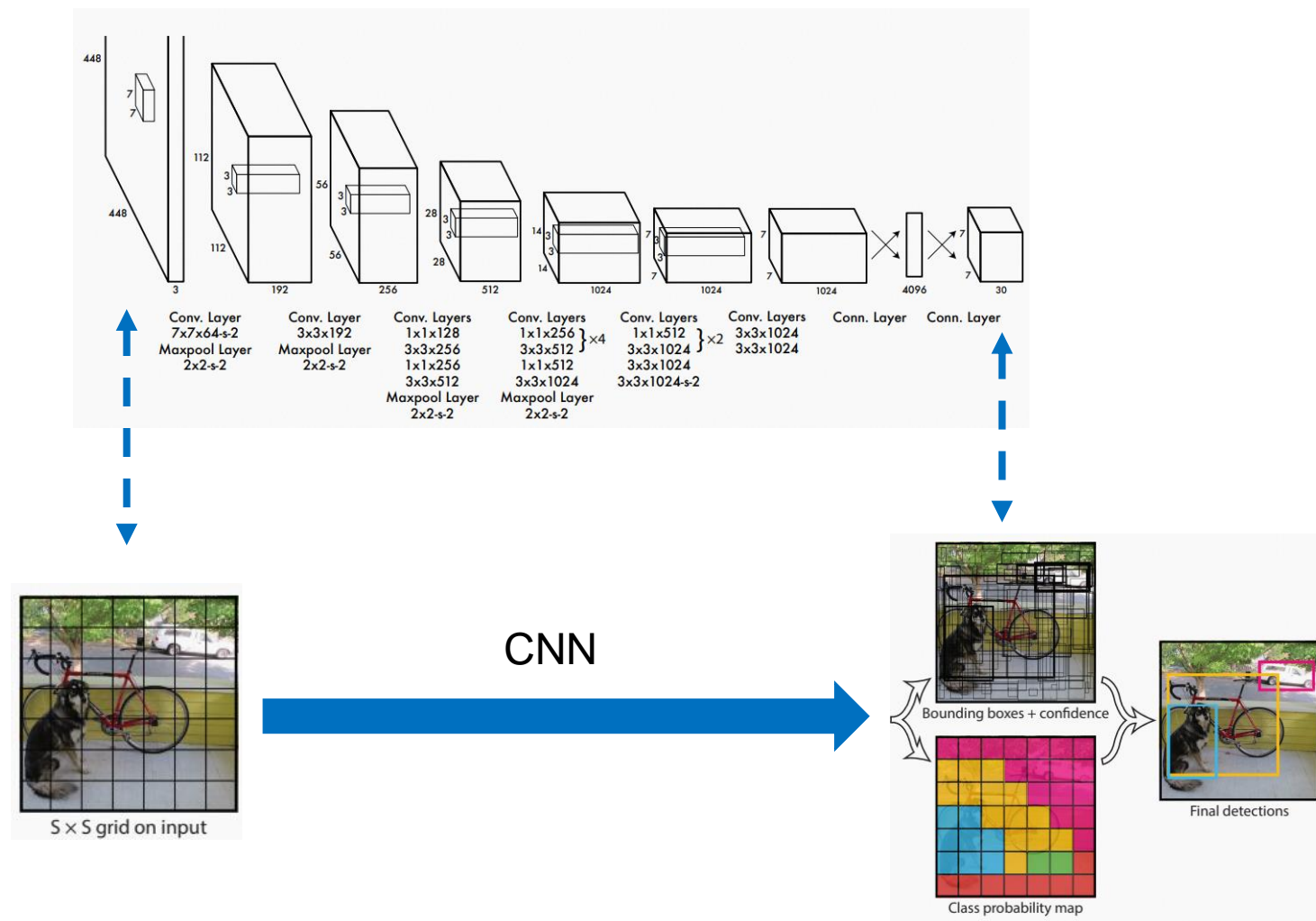
- 使用CNN (Convolution Neural Network) 进行整图 特征提取
- 使用RPN (Region Proposal Network) 进行候选框的提取
- 使用检测头对候选框的位置进行微调, 并得到最终类别和分数
- 优势:
 - 首个端到端的方式, 依靠训练进行网络优化
 - 特征提取效率较高, 使用CNN统一提取特征
 - 精度较高, 使用两阶段的预测方式, 对尺度、小物体比较鲁棒
- 不足:
 - 速度还未达到实时



一阶段目标检测—YOLO

YOLO^[1]:

- 使用CNN (Convolution Neural Network) 进行一次特征提取
- 将原图划分为SxS的网格
- 每个网格进行目标的分类和回归
- 优势：
 - 速度快，无需生成候选框
 - 网络简洁，通用性强
- 不足：
 - 精度有所下降，特别是对小物体



[1] You Only Look Once: Unified, Real-Time Object Detection, CVPR 2016

目标检测新范式—DETR

DETR^[1]:

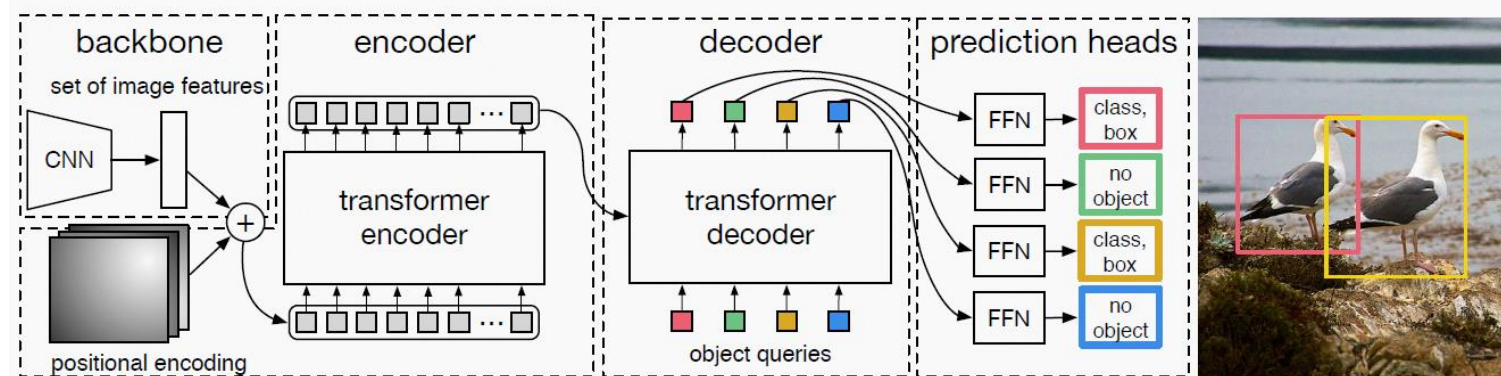
- 引入Transformer结构
- 引入object query, 进行目标预测
- 采用集合预测的思路, 去除NMS, 候选框等以往检测方法常用操作

➤ 优势:

- 目标检测新范式
- 不需要复杂的自定义操作, 实现上更加简单和优雅

➤ 不足:

- 训练时间比较长
- 对小物体检测精度较低



03 总结展望

总结和展望

总结：

1. 了解了图像检测任务的基本知识（任务定义、数据集、评价指标等）
2. 了解了图像检测的代表性方法（Faster-RCNN, YOLO, DETR）

展望：

1. 更细化的方向，如：长尾、零样本、少样本、增量等
2. 更通用的方向，如：SAM^[1], Grounded-SAM^[2]

ModelScope公众号



ModelScope交流群



ModelScope主页



OpenVI对外交流钉钉群



Tips：2024届实习生春招已经开始，并长期招聘研究型实习生。

欢迎联系邮箱： lee.lcy@alibaba-inc.com

[1] Segment Anything, arXiv 2023

[2] <https://github.com/IDEA-Research/Grounded-Segment-Anything>

图像检测实战课程

李晨阳

达摩院-开放视觉智能实验室

ModelScope交流群



OpenVI对外交流钉钉群



课程概览

01 通用检测（李晨阳）

02 人脸检测（刘洋）

03 检测工具箱（蓝劲鹏）

目录&目标

01 ModelScope 图像检测模型介绍

- 了解ModelScope上检测类模型的基本情况

02 创空间实战（安全帽检测）

- 掌握如何搭建一个检测类模型的创空间

03 检测模型实战（安全帽检测）

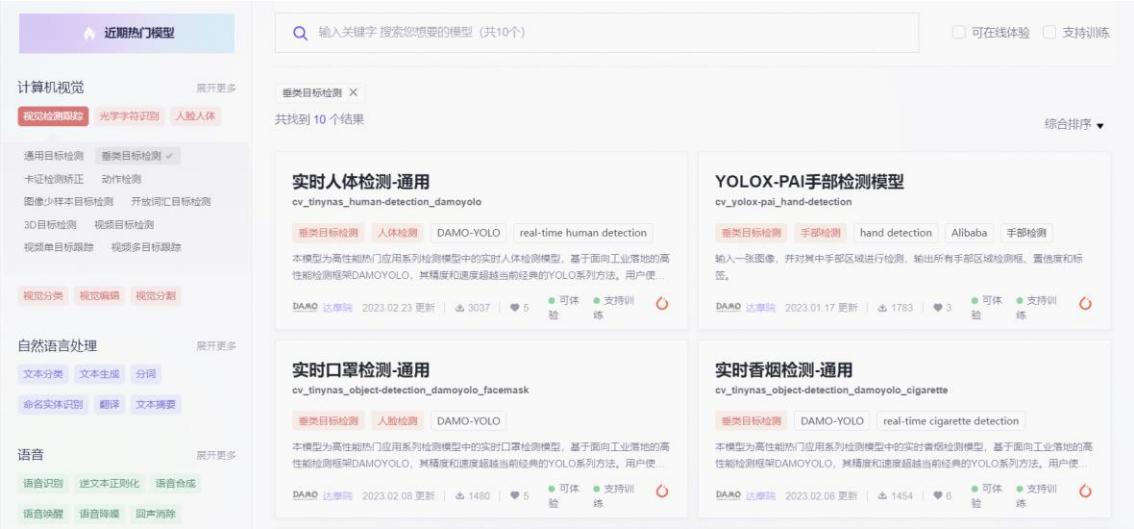
- 掌握如何基于已有的模型进行训练、推理、评估

01 ModelScope 图像检测模型介绍

ModelScope检测模型入口



ModelScope主页



ModelScope泛检测模型归类



图片

视频

其它

通用目标检测

实时目标检测-通用领域

cv_cspnet_image-object-detection_yolox

DINO-高精度目标检测模型

cv_swinl_image-object-detection_dino

VitDet图像目标检测

cv_vit_object-detection_coco

实时目标检测-通用领域-移动端

cv_cspnet_image-object-detection_yolox_nano_coco

DAMOYOLO-高性能通用检测模型-S

cv_tinynas_object-detection_damoyolo

AIRDet-高性能检测模型-S

cv_tinynas_detection

垂类目标检测

实时人体检测-通用

cv_tinynas_human-detection_damoyolo

实时人头检测-通用

cv_tinynas_head-detection_damoyolo

实时口罩检测-通用

cv_tinynas_object-detection_damoyolo_facemask

实时安全帽检测-通用

cv_tinynas_object-detection_damoyolo_safety-helmet

实时目标检测-自动驾驶领域

cv_yolox_image-object-detection-auto

无人机检测模型-S

cv_tinynas_uav-detection_damoyolo

人体检测-通用-Base

cv_resnet18_human-detection

YOLOX-PAI手部检测模型

cv_yolox-pai_hand-detection

实时香烟检测-通用

cv_tinynas_object-detection_damoyolo_cigarette

实时手机检测-通用

cv_tinynas_object-detection_damoyolo_phone

实时交通标识检测-自动驾驶领域

cv_tinynas_object-detection_damoyolo_traffic_sign

实时烟火检测-通用

cv_tinynas_object-detection_damoyolo_smokefire

目标检测

StreamYOLO实时视频目标检测-自动驾驶领域

cv_cspnet_video-object-detection_streamyolo

LongShortNet实时视频目标检测-自动驾驶领域

damo/cv_cspnet_video-object-detection_longshortnet 达摩院提供 | 38次下载

目标跟踪

视频单目标跟踪-通用领域

cv_vitb_video-single-object-tracking_ostrack

ProContEXT视频单目标跟踪-通用领域

cv_vitb_video-single-object-tracking_procontext

Siamfc视频单目标跟踪-无人机-S

cv_alex_video-single-object-tracking_siamfc-uav

视频多目标跟踪-行人

cv_yolov5_video-multi-object-tracking_fairmot

动作检测

日常动作检测

cv_ResNetC3D_action-detection_detection2d

3D目标检测

DEPE-3D目标检测-自动驾驶领域

cv_object-detection-3d_depe

Open World目标检测

基于视觉和语言的知识蒸馏的开放词汇目标检测

cv_resnet152_open-vocabulary-detection_vild

长尾/少样本目标检测

针对长尾/小目标问题的高性能通用目标检测

cv_resnet50_object-detection_maskscoring

defrcn少样本目标检测

cv_resnet101_detection_fewshot-defrcn

显著性、伪装目标检测等

02 创空间实战（安全帽检测）

安全帽检测模型

搭建

创空间应用

创建创空间

准备文件

启动脚本 (app.py)
说明文档 (README.md)

上传文件

发布创空间

ModelScope

[首页](#)[模型库](#)[数据集](#)[创空间](#)[文档中心](#)[动态](#)[社区](#)[GitHub](#)

MorningsunLee

- 贡献者指南
- 模型详解
- 模型库
- 模型部署
- 数据集
- 创空间
 - 创空间介绍
 - 创空间创建与搭建
 - 创空间卡片
 - 创空间示例
- Notebook
- 竞赛
- 组织与个人中心
- 用户协议
- ModelScope社区
- 版本公告

本篇文章介绍ModelScope创空间的产品基本介绍。

关于ModelScope的创空间

ModelScope的创空间（Studio），为您提供自由灵活的AI应用展示空间。您可以基于ModelScope平台上模型提供的原子能力，自行搭建与展示不同AI应用，包括自定义的模型输入输出，多模型的组合，以及可视化交互展现形式等等。我们希望在这里，激发AI开发者的创造力与创新性。由创空间为白纸，以模型为颜料，以想象力为画笔，共谱AI应用五彩斑斓的新画卷！

特别说明：您可以自由创建多个创空间并分享给您的朋友，但考虑到首页的空间展示质量，您所创建的空间并不会自动进入创空间首页精选列表，如您有意愿申请进入首页列表，非常欢迎与我们联系（钉钉群：8010015744，邮箱：contact@modelscope.cn）

名词解释

创空间（Studio）：ModelScope平台提供的模型应用可视化私域空间与运营阵地，您可以在此基于ModelScope上丰富的模型生态，灵活搭建多种多样的AI应用。

Gradio：Gradio提供轻量化的机器学习交互式web页面定制工具，为开发者迅速定制AI应用提供快速上手的脚手架。创空间目前支持通过Gradio来定制围绕ModelScope模型搭建的线上应用。[点击访问Gradio官网](#) 了解更多

Streamlit：Streamlit相较Gradio更为高阶，Streamlit官方提供了更为丰富的示例与交互模板，支持MarkDown与HTML的渲染。创空间目前支持通过Streamlit来定制围绕ModelScope模型搭建的线上应用。[点击访问Streamlit官网](#) 了解更多

Static：ModelScope平台预置的HTML Static SDK，支持快速将已有HTML页面展示在您的项目空间中，适合重交互与视觉体验的算法Demo展示。

快速入门

个人中心 · 魔搭社区

项目空间首页 · 魔搭社区

modelscope.cn/studios?page=1&sort=default&type=0

常用网页 其它网页

ModelScope

首页 模型库 数据集 创空间 文档中心 动态 社区 GitHub

搜索你感兴趣的内容

MorningsunLee

ModelScope创空间

分享创意，探索模型应用的无限想象空间!

了解更多

创建空间

输入基础信息快速完成空间创建

搭建应用

基于平台模型与可视化SDK几行代码自由搭建AI应用

发布空间

平台提供CPU/GPU运行环境支持空间一键发布

分享创意

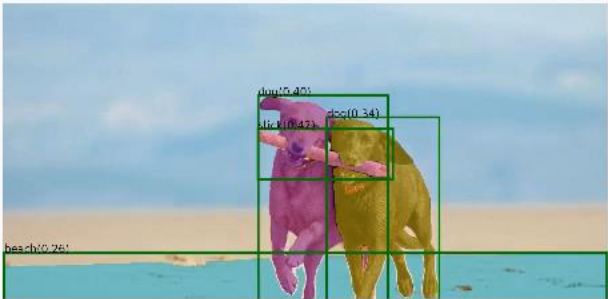
将您的创空间分享给您的朋友吧

精选 全部

输入关键字 搜索您想要的创空间


多模态 计算机视觉 自然语言处理 语音 AutoML

综合排序




Grounded-SAM 马赛克熊

Grounded-SAM把 SAM和 BLIP、Stable Diffusion集成在一起，将图片「分割」、「检测」和「生成」三种能力合一，成



网文小白文模型 RWKV-4-Novel-7B-Ch...

RWKV 语言模型（用纯100%RNN达到GPT能力，甚至更强）。模型介绍：<https://github.com/BlinkDL/RWKV-LM> 在自



模型生成效果评测

提供业内领先的多个类ChatGPT大模型进行效果对比评测。

22:46 2023/4/16

03 检测模型实战（安全帽检测）

推理

inference

定性分析

评估

evaluation

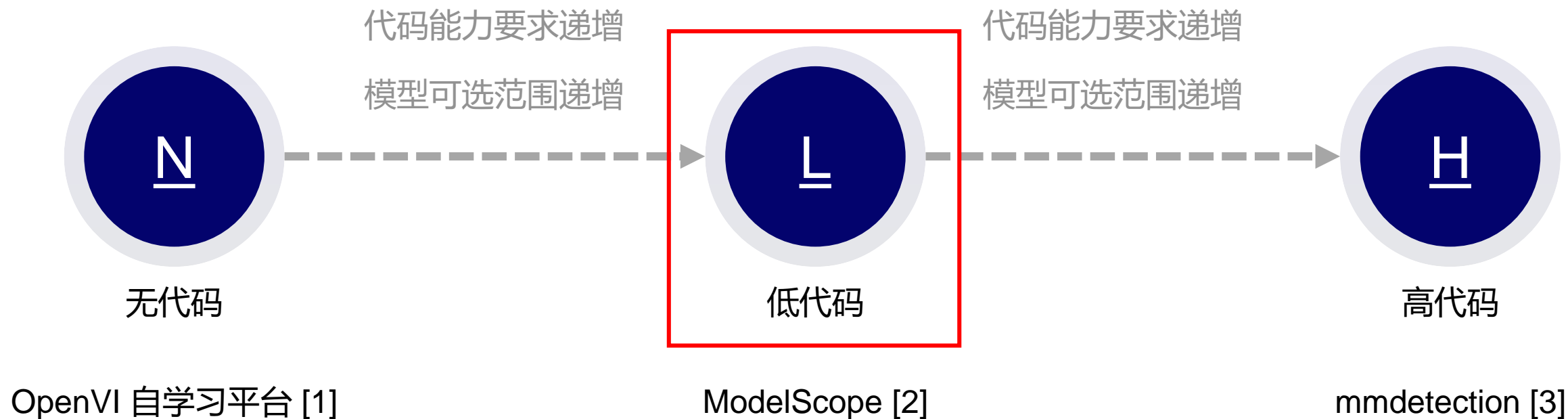
定量分析

训练

train / finetune

精度优化
类别扩充

实战介绍



[1] <https://vision.console.aliyun.com/cn-shanghai/ability/workspace/ws-list>

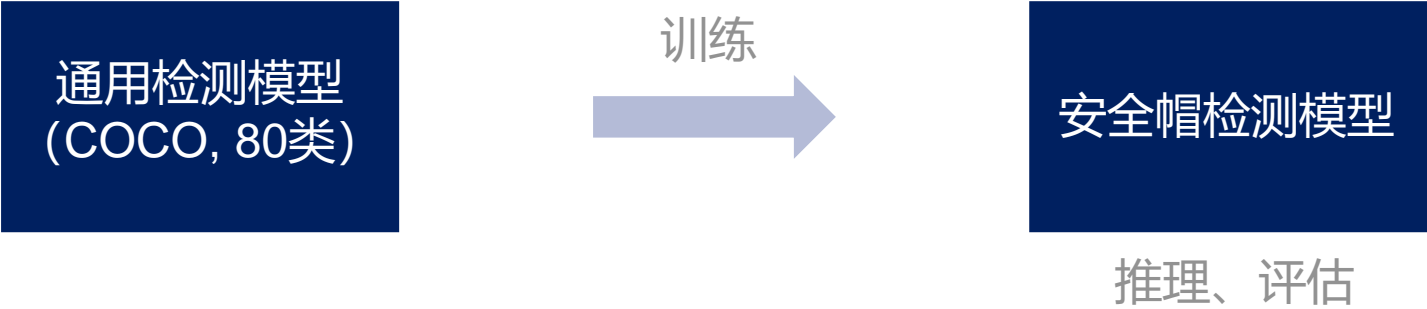
[2] <https://modelscope.cn/home>

[3] <https://github.com/open-mmlab/mmdetection>

检测模型实战



通用检测模型



检测模型实战



通用检测模型
(COCO, 80类)

训练



安全帽检测模型

推理、评估

- 课后练习：
 - 基于已有的通用检测/垂类检测模型，搭建一个自己的创空间。

ModelScope公众号



ModelScope交流群



ModelScope主页



OpenVI对外交流钉钉群



Tips: 2024届实习生春招已经开始，并长期招聘研究型实习生。

欢迎联系邮箱: lee.lcy@alibaba-inc.com

Thanks

附录



app.py

```
# 导入相关模块
import gradio as gr
from modelscope.pipelines import pipeline
from modelscope.utils.constant import Tasks
from modelscope.outputs import OutputKeys
import cv2
import numpy as np
import os

# 声明安全帽检测器
detector = pipeline(Tasks.domain_specific_object_detection, model='damo/cv_tinydas_object-detection_damoyolo_safety-helmet')

# 可视化函数
def show_image_object_detection_auto_result(img, detection_result):
    scores = detection_result[OutputKeys.SCORES]
    labels = detection_result[OutputKeys.LABELS]
    bboxes = detection_result[OutputKeys.BOXES]
    assert img is not None, f"Image is None!!!"

    for (score, label, box) in zip(scores, labels, bboxes):
        cv2.rectangle(img, (int(box[0]), int(box[1])),
                       (int(box[2]), int(box[3])), (0, 0, 255), 2)
        cv2.putText(
            img,
            f'{score:.2f}', (int(box[0]), int(box[1])),
            1,
            1.0, (0, 255, 0),
            thickness=1,
            lineType=8)
        cv2.putText(
            img,
            label, (int(box[0]), int(box[3])),
            1,
            1.0, (0, 255, 0),
            thickness=1,
            lineType=8)

    return img

# 安全帽检测
def safety_helmet_detect(image):
    result = detector(image[...,:-1]) # RGB to BGR
    output_image = show_image_object_detection_auto_result(image, result)
    return output_image

# 创建gradio的interface
title = "安全帽检测"
description = "上传图片，可自动判断图中的人是否佩戴安全帽。"
examples = [['./part2_000125.jpg'], ]
demo = gr.Interface(fn=safety_helmet_detect, inputs=["image"], outputs=[gr.Image(label="检测后图片")], examples=examples, title=title, description=description)
demo.launch()
```

附录



README.md.

```
---
domain:
- cv
tags:
- safety_helmet_detection
- 安全帽检测
models:
- damo/cv_tinytas_object-detection_damoyolo_safety-helmet
license: Apache License 2.0
deployspec:
  entry_file: app.py
---
#### Clone with HTTP
```bash
git clone https://www.modelscope.cn/studios/damo/cv_image_gathering_dection.git
```

# 检测模型实战



## 开启notebook



通用检测模型

## 模块引入

```
模块引入
import os
from modelscope.metainfo import Trainers
from modelscope.trainers import build_trainer
from modelscope.msdatasets import MsDataset
from modelscope.utils.constant import DownloadMode
from modelscope.hub.snapshot_download import snapshot_download
```

## 数据集下载

```
数据集准备
train_dataset = MsDataset.load(
 'head_detection_for_train_tutorial',
 namespace="modelscope",
 split='train',
 download_mode=DownloadMode.REUSE_DATASET_IF_EXISTS)
val_dataset = MsDataset.load(
 'head_detection_for_train_tutorial',
 namespace="modelscope",
 split='validation',
 download_mode=DownloadMode.REUSE_DATASET_IF_EXISTS)
```



# 检测模型实战

## 相关参数设置

```
相关参数设置
model_id = 'damo/cv_cspnet_image-object-detection_yolox'
work_dir = './output/cv_cspnet_image-object-detection_yolox_1'
trainer_name = Trainers.easydcv
batchsize = 8
total_epochs = 15
class_names = ['head']

cfg_options = {
 'train.max_epochs':
 total_epochs,
 'train.dataloader.batch_size_per_gpu':
 batchsize,
 'evaluation.dataloader.batch_size_per_gpu':
 1,
 'train.optimizer.lr': 0.01 / 64 * batchsize,
 'train.lr_scheduler.warmup_iters': 1,
 'train.lr_scheduler.num_last_epochs': 5,
 'train.hooks': [
 {
 'type': 'CheckpointHook',
 'interval': 5
 },
 {
 'type': 'EvaluationHook',
 'interval': 5
 },
 {
 'type': 'TextLoggerHook',
 'ignore_rounding_keys': None,
 'interval': 1
 }
],
 'dataset.train.data_source.classes': class_names,
 'dataset.val.data_source.classes': class_names,
 'evaluation.metrics.evaluators': [
 {
 'type': 'CocoDetectionEvaluator',
 'classes': class_names
 }
],
 'CLASSES': class_names,
}
```

## 开启训练

```
新建trainer
kwargs = dict(
 model=model_id,
 train_dataset=train_dataset,
 eval_dataset=val_dataset,
 work_dir=work_dir,
 cfg_options=cfg_options)
print("build trainer.")
trainer = build_trainer(trainer_name, kwargs)
print("start training.")

预训练模型下载
cache_path = snapshot_download(model_id)

开启训练
trainer.train(
 checkpoint_path=os.path.join(cache_path, 'pytorch_model.pt'),
 load_all_state=False,
)
```



通用检测模型

## 目录结构

```
root@dsw-22334-f57b47c9-z16tv:/mnt/workspace/tutorial# ls output/cv_cspnet_image-object-detection_yolox_1/
20230416_160626.log epoch_10.pth epoch_15.pth epoch_5.pth output
20230416_160626.log.json epoch_10_trainer_state.pth epoch_15_trainer_state.pth epoch_5_trainer_state.pth
```

```
root@dsw-22334-f57b47c9-z16tv:/mnt/workspace/tutorial# ls output/cv_cspnet_image-object-detection_yolox_1/output/
configuration.json pytorch_model.bin pytorch_model.pt README.md res test_object_0.jpeg
```



# 检测模型实战

## 模型评估

```
模型评估
eval_res = trainer.evaluate(checkpoint_path=os.path.join(work_dir, 'epoch_15.pth'))
print(eval_res)
```

## 模型推理

```
模型推理
from modelscope.pipelines import pipeline
from modelscope.utils.constant import Tasks
from modelscope.outputs import OutputKeys

realtime_detector = pipeline(Tasks.image_object_detection, model=os.path.join(work_dir, 'output'))
result = realtime_detector('https://modelscope.oss-cn-beijing.aliyuncs.com/test/images/image_detection.jpg')
打印 bbox
print(result)
```



通用检测模型

## 结果可视化

```
def vis_det_img(input_path, res):
 def get_color(idx):
 idx = (idx + 1) * 3
 color = ((10 * idx) % 255, (20 * idx) % 255, (30 * idx) % 255)
 return color
 img = cv2.imread(input_path)
 unique_label = list(set(res['labels']))
 for idx in range(len(res['scores'])):
 x1, y1, x2, y2 = res['boxes'][idx]
 score = str(res['scores'][idx])
 label = str(res['labels'][idx])
 color = get_color(unique_label.index(label))
 cv2.rectangle(img, (int(x1), int(y1)), (int(x2), int(y2)), color, 2)
 cv2.putText(img, label, (int(x1), int(y1) - 10),
 cv2.FONT_HERSHEY_PLAIN, 1, color)
 cv2.putText(img, score, (int(x1), int(y2) + 10),
 cv2.FONT_HERSHEY_PLAIN, 1, color)
 return img

im_vis = vis_det_img('./image_detection.jpg', result)
cv2.imwrite('./image_detection_vis.jpg', im_vis)
```