

Object Detection Framework: MMDetection

— A Survey on Object Detection

汇报人：孙寒石

其他组员：张扬 朱星宇

School of Electronic Science & Engineering
Southeast University

preminstrel@seu.edu.cn

June 14, 2022



東南大學
SOUTHEAST UNIVERSITY

Presentation Outline

① Introduction

② MMDetection

③ Demo

④ Resource

1 Introduction

2 MMDetection

3 Demo

4 Resource

Introduction to Object Detection

目标检测 (Object Detection) 的任务是找出图像中所有特定的目标，确定它们的类别和位置。由于各类物体有不同的外观和形状，加上成像时光照、遮挡等因素的干扰，目标检测一直是 CV 领域内具有挑战性的问题。

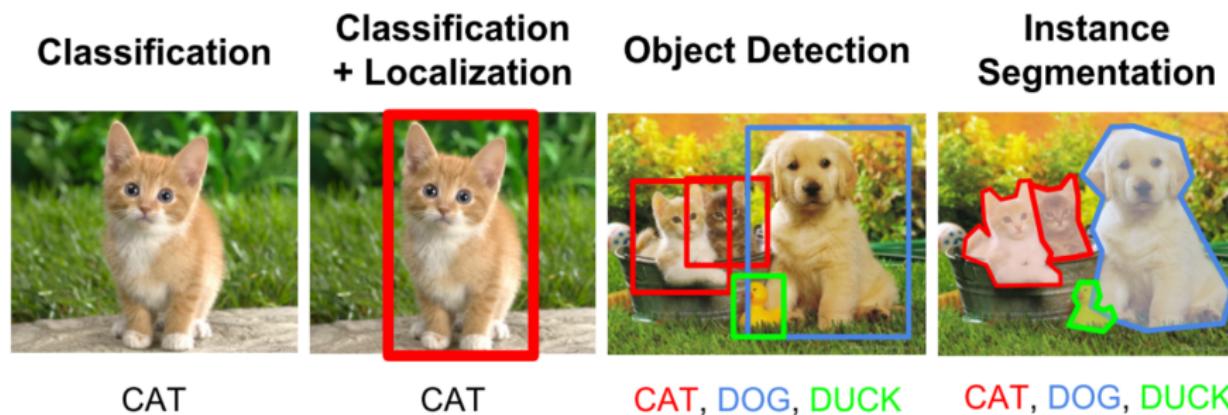


Figure 1: Object Detection

Introduction to Featured Algorithms

基于深度学习的目标检测算法主要分为两类：**Two stage** 和 **One stage**。

- 常见 **two stage** 算法有：R-CNN、Fast R-CNN、Faster R-CNN 等。
- 常见 **one stage** 算法有：YOLO、SSD 和 RetinaNet 等。
- 最近从 NLP 迁移的 **Transformer**，也是包括于前两种，如 DETR。
算法这里就不多做介绍了，都写在了课程报告里 (Page. 4 - 30)。
个人比较喜欢 DETR，因为它省略了 hand-crafted detection pipeline.

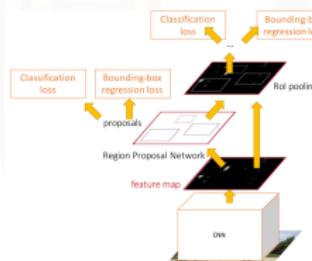


Figure 2: Faster R-CNN

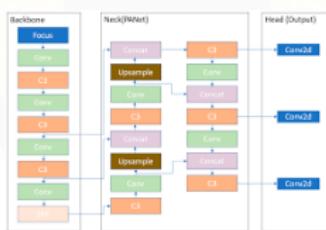


Figure 3: YOLO

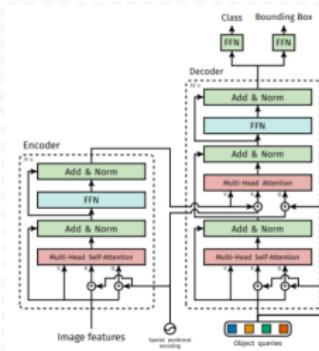


Figure 4: DETR

Introduction to MMDetection

MMDetection [1] 是一个基于 **PyTorch** 的目标检测开源工具箱，属于 OpenMMLab 项目。目前已复现大部分主流/前沿模型，例如 Faster R-CNN [2]、YOLO 和较新的 DETR[3] 等，模型库十分丰富，社区维护氛围较好 ( open-mmlab/mmdetection ⭐19.7k)。在学术研究和工业落地中应用非常广泛。

- **PyTorch** ：报告中简略介绍了原理和语法 ([Page. 35 - 44](#))。
- **Datasets**：报告中介绍了一些经典目标检测数据集 ([Page. 31 - 34](#))。



Figure 5: MMDetection

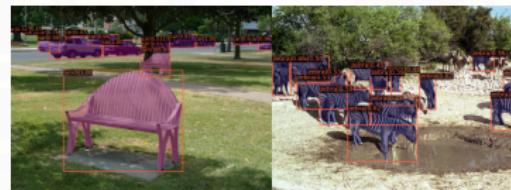


Figure 6: MMDetection Demo

① Introduction

② MMDetection

③ Demo

④ Resource

Framework of MMDetection

MMDetection 将检测拆解模块化为 **Backbone**, **Neck**, **Head** 等。线索清晰, 体系自成。

基于目前代码实现, 所有目标检测算法都按照 Fig. 7 流程进行划分。
任何一个 batch 的图片先输入到 **Backbone** 中进行特征提取, 输出的特征图输入到 **Neck** 模块中进行特征融合或者增强, 最终输入到 **Head** 部分, 一般都会包括分类和回归分支输出。

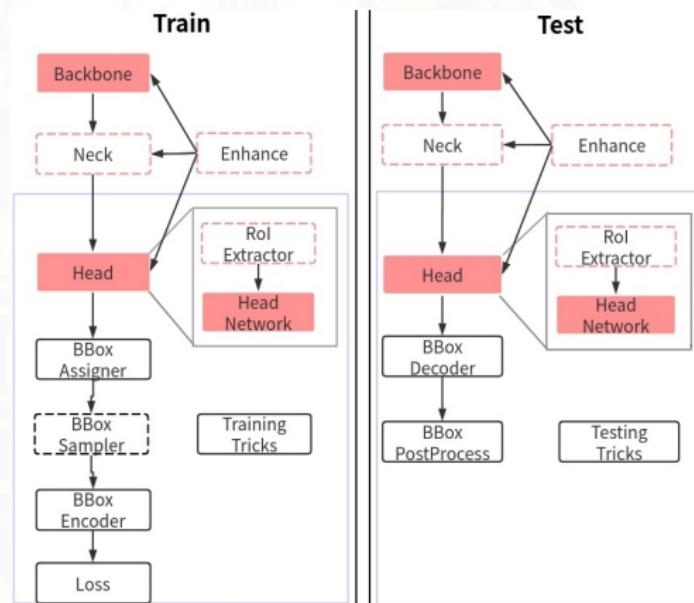


Figure 7: Overview

Backbone

Backbone 可以对图片进行**特征提取**, 比如著名的 ResNet。目前 MMDetection 中已集成了大部分骨架网络, 具体见: `mmdet/models/backbones`。

```
1 | __all__ = [
2 |     'RegNet', 'ResNet', 'ResNetV1d', 'ResNeXt', 'SSDVGG', 'HRNet',
3 |     'Res2Net', 'HourglassNet', 'DetectoRS_ResNet', 'DetectoRS_ResNeXt',
4 |     'Darknet', 'ResNeSt', 'TridentResNet'
5 | ]
```

若要对 Backbone 进行扩展, 可继承上述网络, 用注册器机制注册使用。

```
1 | pretrained='torchvision://resnet50',
2 | backbone=dict(type='ResNet', # 骨架类名, 初始化参数 depth=50,
3 |     num_stages=4,
4 |     out_indices=(0, 1, 2, 3),
5 |     frozen_stages=1,
6 |     norm_cfg=dict(type='BN', requires_grad=True),
7 |     norm_eval=True,
8 |     style='pytorch'),
```



Neck

Neck 是 Backbone 和 Head 的连接层，可对 Backbone 的特征进行高效融合和增强，能够对输入的单尺度或者多尺度特征进行融合、增强输出等。具体见：`mmdet/models/necks`。

```
1 | __all__ = [
2 |     'FPN', 'BFP', 'ChannelMapper', 'HRFPN', 'NASFPN', 'FPN_CARAFE',
3 |     'PAFPN', 'NASFCOS_FPN', 'RFP', 'YOLOV3Neck'
4 | ]
```

最常用的应该是 FPN，一个典型用法是：

```
1 | neck=dict(
2 |     type='FPN',
3 |     in_channels=[256, 512, 1024, 2048], # 骨架多尺度特征图输出通道
4 |     out_channels=256, # 增强后通道输出
5 |     num_outs=5), # 输出 num_outs 个多尺度特征图
```

Head

目标检测算法输出一般包括分类和框坐标回归两个分支，不同算法 Head 模块复杂程度不一样，灵活度比较高。在网络构建方面，理解目标检测算法主要是要理解 Head 模块。

```
1  __all__ = [
2      'AnchorFreeHead', 'AnchorHead', 'GuidedAnchorHead', 'RPNHead',
3      'GARPNHead', 'RetinaHead', 'RetinaSepBNHead', 'GARetinaHead',
4      'SSDHead', 'FCOSHead', 'RepPointsHead', 'FoveaHead',
5      'FreeAnchorRetinaHead', 'ATSSHead', 'FSAFHead', 'NASFCOSHead',
6      'PISARetinaHead', 'PISASSDHead', 'GFLHead', 'CornerHead',
7      'YOLACTHead', 'YOLACTSegmHead', 'YOLACTProtonet', 'YOLOV3Head',
8      'PAAHead', 'SABLRetinaHead', 'CentripetalHead', 'VFNetHead',
9      'FeatureAdaption', 'TransformerHead'
10     ]
```

由于时间限制，MMDetection 剩余的模块（例如 Runner, Hooks 等）不再展开介绍，在课程报告中可以找到相应的讲解（[Page. 66 - 70](#)）。

① Introduction

② MMDetection

③ Demo

④ Resource

Environment Set Up

在做项目的时候，我和导师（9 系 PALM Lab 的一位教授）要来 8 张 Tesla V100 (每张显存 32GB) 来跑，得出这里的 Demo 结果。

如果有同学想复现但没有足够的计算资源的适合，可以利用 [Google Colab](#) 和 [Kaggle](#) 提供的免费算力，是可以用到一张 Tesla P100 的，而且各种依赖和 git repo 的下载速度都很理想。

```
1 from google.colab import drive
2 import os
3
4 drive.mount('/content/gdrive')
5 os.chdir('/content/gdrive/MyDrive/{TARGET FOLDER NAME}/')
```

Demo: Detection on Fundus Images

这个 Demo 利用了 MMDetection 框架来复现一些论文的 Baseline，项目组做的是眼底多模态的课题，选取眼底荧光造影数据集 FFA-IR [4]。

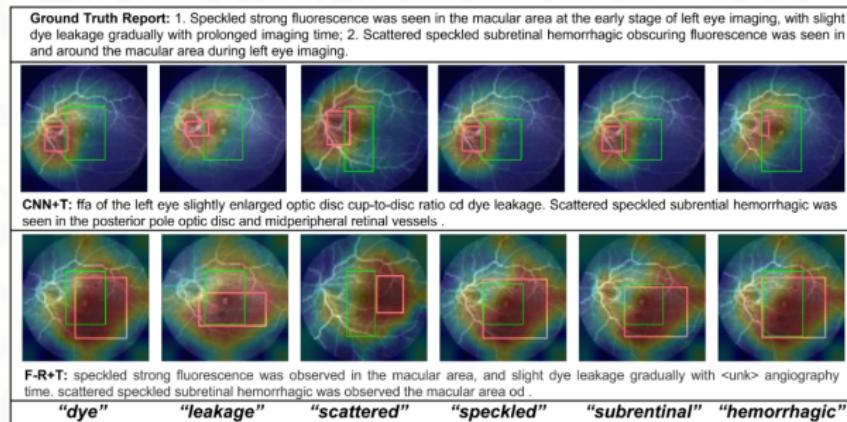


Figure 8: Fundus Images: FFA-IR

首先要对官方给的 raw annotations 的 json 文件进行一个预处理，将标注格式转化为 COCO 格式 [5] 后用 COCODataset 类来加载数据并进行训练以及测试。

Demo: Detection on Fundus Images (Cont'n)

调用 MMDetection 的 config 文件和模块，对经典的网络进行训练（下面列出了部分结果），平均一个网络训练耗时 16 小时（Transformer 收敛要更多时间），选取 COCO 的 mAP 标准在测试集上进行衡量。

$$\text{AP} = \sum_{i=1}^{n-1} (r_{i+1} - r_i) P_{\text{inter}} (r_i + 1) \quad \text{mAP} = \frac{\sum_{i=1}^k AP_i}{k}$$

Metrics	Faster R-CNN	Cascade R-CNN	DETR	Def-DETR
(AP) @[IoU=0.50:0.95 area= all]	0.523	0.543	0.542	0.568
(AP) @[IoU=0.50 area= all]	0.842	0.817	0.796	0.842
(AP) @[IoU=0.75 area= all]	0.576	0.597	0.6	0.623
(AP) @[IoU=0.50:0.95 area= small]	0.479	0.5	0.485	0.461
(AP) @[IoU=0.50:0.95 area=medium]	0.474	0.487	0.443	0.465
(AP) @[IoU=0.50:0.95 area= large]	0.554	0.574	0.574	0.601
(AR) @[IoU=0.50:0.95 area= all]	0.584	0.62	0.644	0.658
(AR) @[IoU=0.50:0.95 area= all]	0.584	0.62	0.644	0.658
(AR) @[IoU=0.50:0.95 area= all]	0.584	0.62	0.644	0.658
(AR) @[IoU=0.50:0.95 area= small]	0.512	0.62	0.512	0.492
(AR) @[IoU=0.50:0.95 area=medium]	0.527	0.549	0.531	0.557
(AR) @[IoU=0.50:0.95 area= large]	0.609	0.549	0.674	0.677

Table 1: Results for Many Models Based on MMDetection

Demo: Detection on Fundus Images (Cont'n)

可以看到，训练的结果还是很理想的。但是由于数据的长尾分布等原因，所以训练出来的准确率在目标很多（大于 10）的时候表现不是特别理想。这个时候利用 Focal Loss 等方法可以有效缓和这种现象。

$$\mathcal{L}_{Focal} = -(1 - p_t)^\gamma \log(p_t)$$

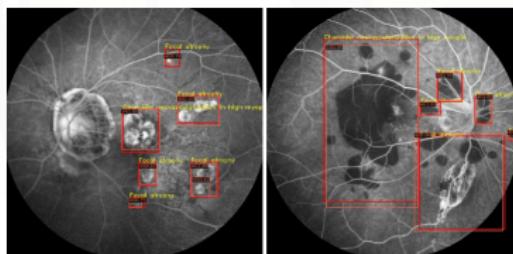


Figure 9: Good Performance

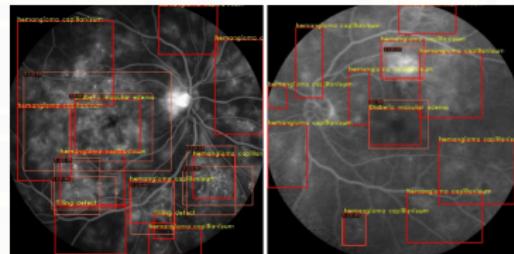


Figure 10: Poor Performance

Demo: *One More Thing * 如果有时间就讲

- 利用 Transformer [6] 的 Attention 机制融合眼底多模态，实现 Multi-task Multimodal 的 Unified-Transformer。

预计投稿 *The British Machine Vision Conference (BMVC 2022)*

- 通过解耦分布学习和一致性正则化对抗医疗噪声标签
已投稿 *FGCS 2022 (SCI, JCR-1 区)*

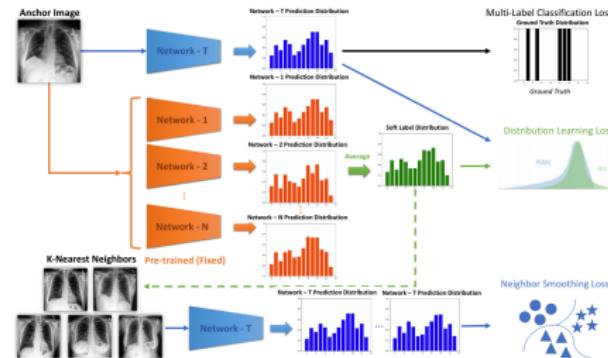


Figure 11: Combating Medical Noisy Labels by Disentangled Distribution Learning and Consistency Regularization [7]

① Introduction

② MMDetection

③ Demo

④ Resource

Resource

课程结束后，课程报告/实验代码等资料将都会上传到 GitHub Repo [preminstrel/computer-vision-2022-project](#)

Computer Vision 2022 Project

[Overview](#) [GitHub](#) [Notes](#)



COMPUTER VISION 2022 PROJECT

Project Team

[Hanshi Sun \(孙寒石\)](#), Project Leader

[Yang Zhang \(张扬\)](#), Project Member

[Xingyu Zhu \(朱星宇\)](#), Project Member

Southeast University, China

Supervisor

[Jian Lu \(陆建\)](#)

Figure 12: Website

References

- [1] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, “MMDetection: Open MMLab Detection Toolbox and Benchmark,”, Jun. 2019.
- [2] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.
- [3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229.
- [4] M. Li, W. Cai, R. Liu, Y. Weng, X. Zhao, C. Wang, X. Chen, Z. Liu, C. Pan, and M. Li, “FFA-IR: Towards an Explainable and Reliable Medical Report Generation Benchmark,” in *Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [5] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, and Y. Xu, “A survey on vision transformer,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- [6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, \. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017, pp. 5998–6008.
- [7] Y. Zhou, L. Huang, T. Zhou, and L. Shao, “Many-to-One Distribution Learning and K-Nearest Neighbor Smoothing for Thoracic Disease Identification,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, ASSOC ADVANCEMENT ARTIFICIAL INTELLIGENCE, 2021, pp. 768–776.

Fin.

Thank You.