Fast_Reid使用教程

FastReid_QuickStart

Mainly explain how to customize the dataset, modify the skeleton, modify various parameter methods, etc

项目链接:JDAI-CV/fast-reid: SOTA Re-identification Methods and Toolbox

主要讲解如何自定义数据集, 修改骨架, 修改各种参数方法等等

自定义数据集

配置好环境, 首先跑通官方数据集.....

测试Market1501数据集

按照官方doc, 下载数据集(或者自己搜索下载), 放到合适的地方
测试使用market1501数据集运行

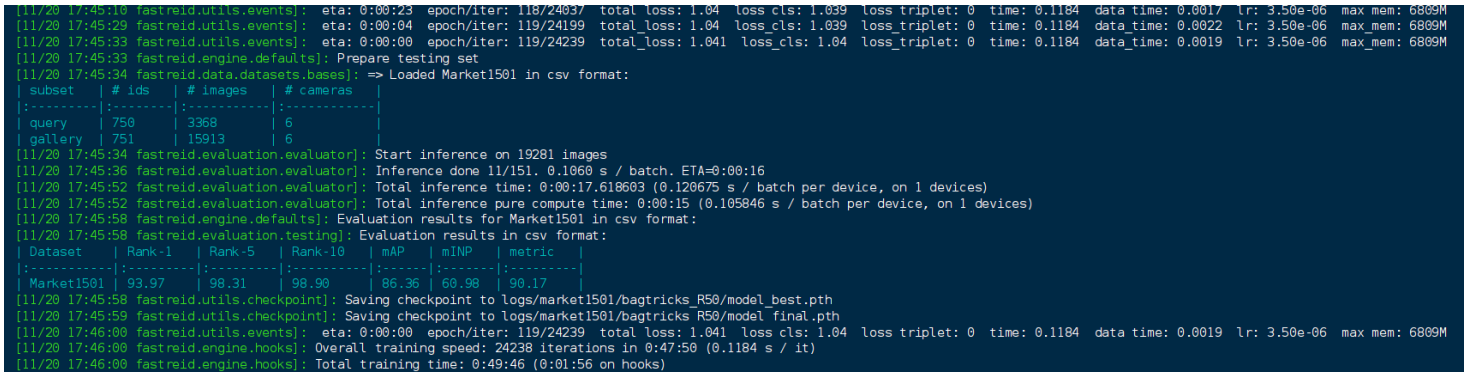
```
python3 tools/train_net.py --config-file ./configs/Market1501/bagtricks_R50.yml
MODEL.DEVICE "cuda:0"
```

评估模型的性能，使用

```
python3 tools/train_net.py --config-file ./configs/Market1501/bagtricks_R50.yml --
eval-only \
MODEL.WEIGHTS /path/to/checkpoint_file MODEL.DEVICE "cuda:0"
```

更多选项，参阅 `python3 tools/train_net.py -h`。

然后这个测试的结果是:



测试成功

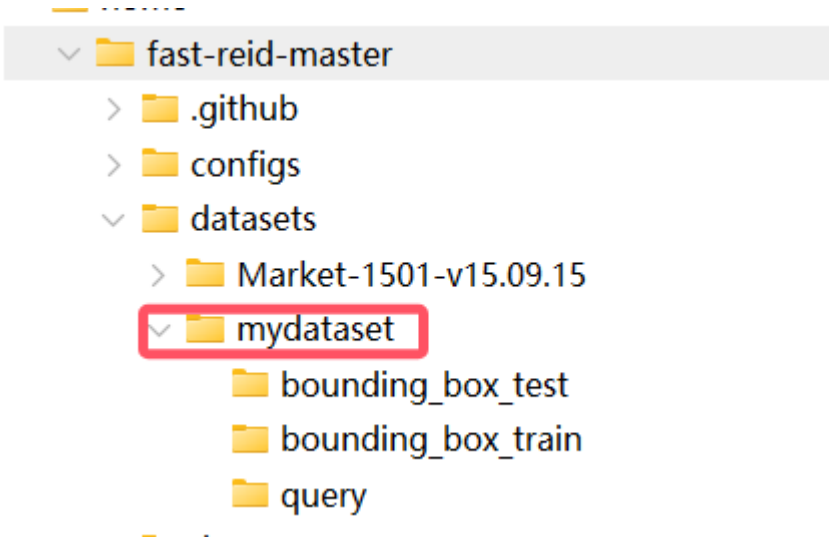
加载自己的数据集

接下来试试怎么加载自己的数据集.....

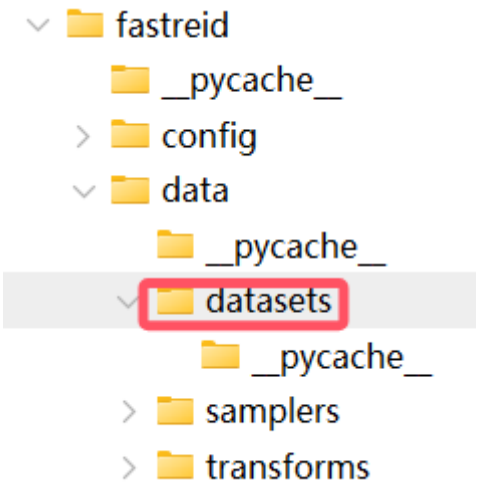
需要更改的地方

在dataset中创建自己的数据集, 这里我取名为 `mydataset`

注意, 目录结构和格式可以参考Market1501数据集的各个配置

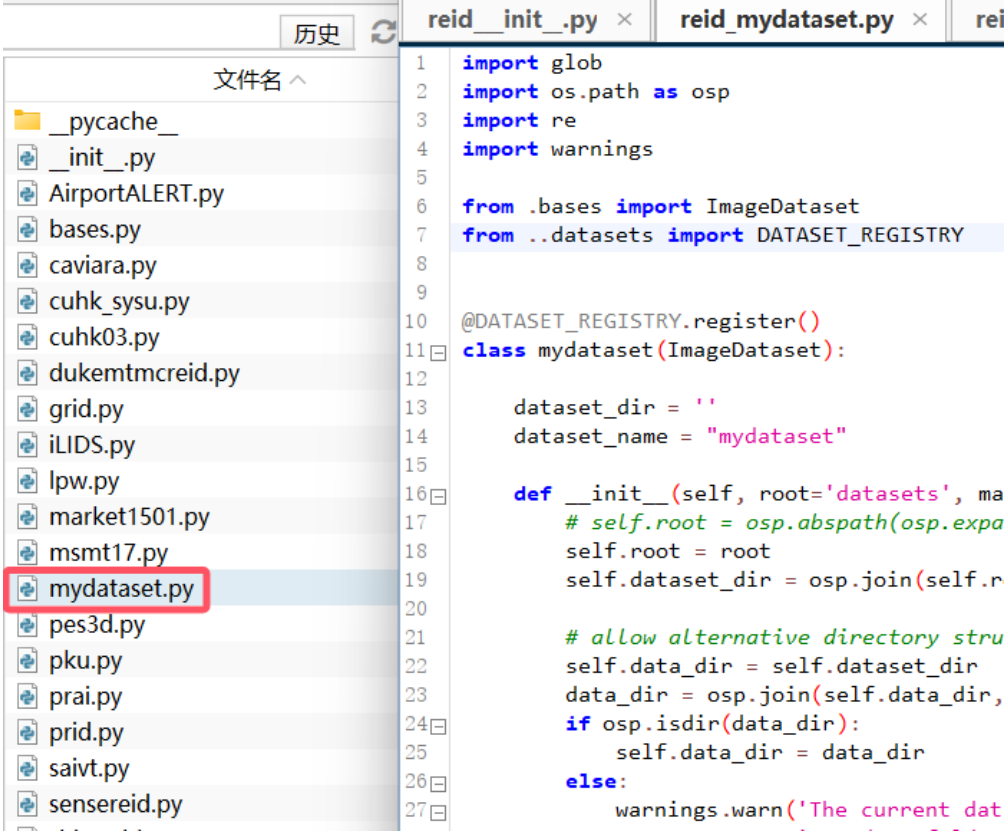


图片的命名格式和配置的目录结构名称在fastreid/data/datasets/目录下修改配置文件修改



下面介绍这个目录下的配置

依然参考上面market1501的配置代码, 详细读一下, 就知道如何修改了
(根据自己的需求修改)



格式的修改:
按照这个类的框架来写:

```
from fastreid.data.datasets import DATASET_REGISTRY
from fastreid.data.datasets.bases import ImageDataset


@DATASET_REGISTRY.register()
class MyOwnDataset(ImageDataset):
        def __init__(self, root='datasets', **kwargs):
                '''
                中间是自己的处理代码
                主要逻辑是由自己的数据集组成符合格式要求的元组列表
                只需要关注_init_的传入即可
                '''
                super().__init__(train, query, gallery)
```

作者要求super的传入格式:

训练列表：`list(str, str, str)`,
查询列表：`list(str, int, int)` ,
图库列表：`list(str, int, int)`。

```
train_list = [
(train_path1, pid1, camid1), (train_path2, pid2, camid2), ...]

query_list = [
(query_path1, pid1, camid1), (query_path2, pid2, camid2), ...]
```

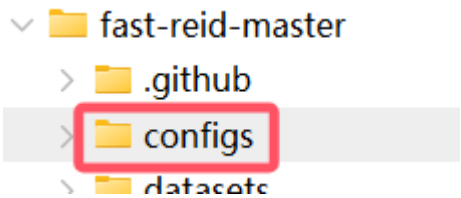
```
gallery_list = [
(gallery_path1, pid1, camid1), (gallery_path2, pid2, camid2), ...]
```

数据格式参考官方文档

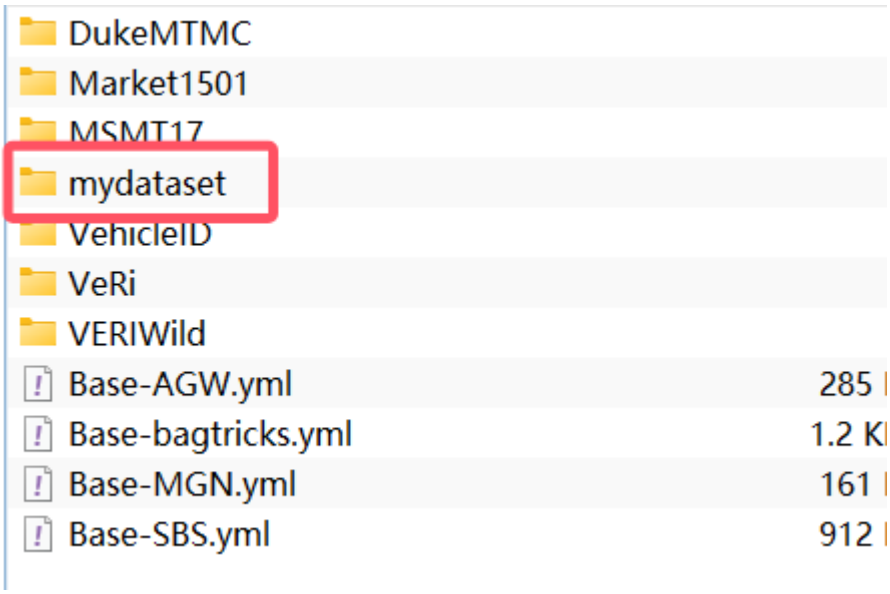
创建好自己的数据集类之后
在当前目录中的 `_init.py` 中引入:

```python
from ...utils.registry import Registry

DATASET_REGISTRY = Registry("DATASET")
DATASET_REGISTRY.__doc__ = """
Registry for datasets
It must returns an instance of :class:`Backbone`.
"""

# Person re-id datasets
from .mydataset import mydataset
# my dataset

from .cuhk03 import CUHK03
```

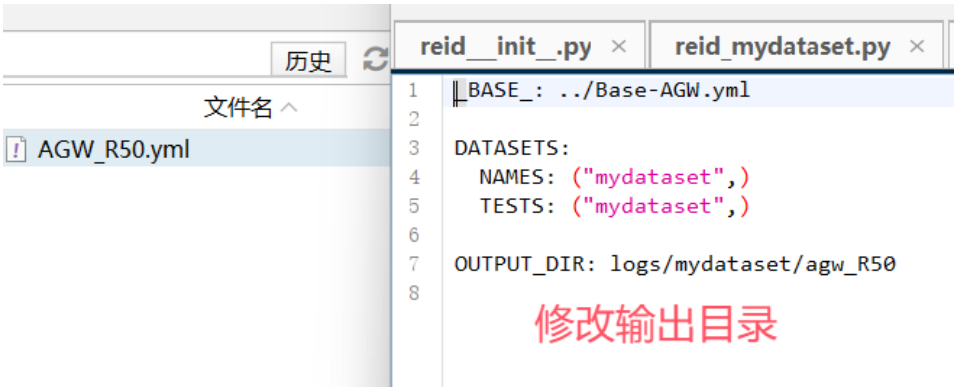
最后一步,修改Configs配置文件



创建自定义数据集配置文件夹:



同样的,参考其他的格式修改对应内容即可, 仿照制作对应的配置文件



> 需要理解的地方: 总体的配置文件是由第一行的 _BASE_ 叠加起来的 一级一级查看文件就能明白

最后在 `tools/train_net.py` 中导入以使其生效

```
from fastreid.config import get_cfg
from fastreid.engine import DefaultTrainer, default_argument_pars
from fastreid.utils.checkpoint import Checkpointer
from fastreid.data.datasets.mydataset import mydataset

def setup(args):
    """
    Create configs and perform basic setups.
    """
```

大致的流程就是这个样子
根据自己的想法修改即可

参考资料

[如何训练自定义数据集 · Issue #220 · JDAI-CV/fast-reid --- How to train Custom Dataset · Issue #220 · JDAI-CV/fast-reid]
[fast-reid跑通自己的数据_fastreid训练自己的数据-CSDN博客]
[FastReID使用教程、踩坑记录 - 知乎]
[Transformer-based Object Re-Identification - 重识别论文阅读 - 行人重识别 - Yifeng's Blog]

以下是采用不同backbone的训练命令

resnet50:

```
python3 tools/train_net.py --config-file ./configs/mydataset/baseline_R50.yml
MODEL.DEVICE "cuda:0"
```

resnet101:

```
python3 tools/train_net.py --config-file ./configs/mydataset/baseline_R101.yml
MODEL.DEVICE "cuda:0"
```

/home/fast-reid-master/configs/mydataset/tricks/config_triplet_loss_weight.yml
vit:

```
python3 tools/train_net.py --config-file ./configs/mydataset/baseline_Vit.yml
MODEL.DEVICE "cuda:0"
```

剩下的等后面补吧.... 没时间写了

增加方法

修改骨架网络结构

这里以ResNet50为例