# 聊聊Pytorch中的dataloader

CV开发者都爱看的 极市平台 2021-07-03 23:30:00 手机阅读





作者丨Mario@知乎（已授权）

来源丨https://zhuanlan.zhihu.com/p/117270644

编辑丨极市平台

## 极市导读

本文介绍了pytorch的dataloader各个参数的含义，并针对num_workers，sample和collate_fn分别进行了详细的说明。 >>加入极市CV技术交流群，走在计算机视觉的最前沿

今天为啥突然要写一下pytorch的dataloader呢，首先来说说事情的来龙去脉。

起初，我最开始单独训练一个网络来完成landmark点回归任务和分类任务，训练的数据是txt格式，在训练之前对数据进行分析，发现分类任务中存在严重的数据样本不均衡的问题，那么我事先针对性的进行数据采样均衡操作，重新得到训练和测试的txt数据和标签，保证了整个训练和测试数据的样本均衡性。由于我的整个项目是检测+点回归+分类，起初检测和点回归+分类是分两步实现的，检测是通过读取XML格式来进行训练，现在要统一整个项目的训练和测试过程，要将点回归+分类的训练测试过程也按照读取XML格式来进行，那么就遇到一个问题，如何针对性的去给样本偏少的样本进行均衡，由于在dataset类中，返回的图像和标签都是针对每个index返回一个结果，在dataset类中进行操作似乎不太可行，那么就想到在dataloader中进行操作，通过dataloader中的参数sample来完成针对性采样。

还有一个问题是关于num_workers的设置，因为我有对比过，在我的单机RTX 2080Ti上和八卡服务器TITAN RTX上(仅使用单卡，其它卡有在跑其它任务)，使用相同的num_workers，在单机上的训练速度反而更快，于是猜想可能和CPU或者内存有关系，下面会具体分析。

首先来看下下dataloader中的各个参数的含义。

类的定义为： `torch.utils.data.DataLoader` ，其中包含的参数有：

```
torch.utils.data.DataLoader(dataset, batch_size=1, shuffle=False, sampler=
    batch_sampler=None, num_workers=0, collate_fn=None, pin_memory=False,
    drop_last=False, timeout=0, worker_init_fn=None, multiprocessing_conte
```

dataset：定义的dataset类返回的结果。

batchsize：每个bacth要加载的样本数，默认为1。

shuffle：在每个epoch中对整个数据集data进行shuffle重排，默认为False。

sample：定义从数据集中加载数据所采用的策略，如果指定的话，shuffle必须为False；batch_sample类似，表示一次返回一个batch的index。

num_workers：表示开启多少个线程数去加载你的数据，默认为0，代表只使用主进程。

collate_fn：表示合并样本列表以形成小批量的Tensor对象。

pin_memory：表示要将load进来的数据是否要拷贝到pin_memory区中，其表示生成的Tensor数据是属于内存中的锁页内存区，这样将Tensor数据转义到GPU中速度就会快一些，默认为False。

drop_last：当你的整个数据长度不能够整除你的batchsize，选择是否要丢弃最后一个不完整的batch，默认为False。

注：这里简单科普下pin_memory，通常情况下，数据在内存中要么以锁页的方式存在，要么保存在虚拟内存(磁盘)中，设置为True后，数据直接保存在锁页内存中，后续直接传入cuda；否则需要先从虚拟内存中传入锁页内存中，再传入cuda，这样就比较耗时了，但是对于内存的大小要求比较高。

下面针对num_workers，sample和collate_fn分别进行说明：

**1. 设置num_workers：**

pytorch中dataloader一次性创建 `num_workers` 个子线程，然后用 `batch_sampler` 将指定batch分配给指定worker，worker将它负责的batch加载进RAM，dataloader就可以直接从RAM中找本轮迭代要用的batch。如果 `num_worker` 设置得大，好处是寻batch速度快，因为下一轮迭代的batch很可能在上一轮/上上一轮...迭代时已经加载好了。坏处是**内存开销大**，也**加**

**重了CPU负担**（worker加载数据到RAM的进程是进行CPU复制）。如果num_worker设为0，意味着每一轮迭代时，dataloader不再有自主加载数据到RAM这一步骤，只有当你需要的时候再加载相应的batch，当然速度就更慢。 num_workers 的经验设置值是自己电脑/服务器的CPU核心数，如果CPU很强、RAM也很充足，就可以设置得更大些，对于单机来说，单跑一个任务的话，直接设置为CPU的核心数最好。

### 2. 定义sample：（假设dataset类返回的是：data, label）

```
from torch.utils.data.sampler import WeightedRandomSampler
## 如果label为1，那么对应的该类别被取出来的概率是另外一个类别的2倍
weights = [2 if label == 1 else 1 for data, label in dataset]
sampler = WeightedRandomSampler(weights,num_samples=10, replacement=True)
dataloader = DataLoader(dataset, batch_size=16, sampler=sampler)
```

PyTorch中提供的这个sampler模块，用来对数据进行采样。默认采用SequentialSampler，它会按顺序一个一个进行采样。常用的有随机采样器：RandomSampler，当dataloader的shuffle参数为True时，系统会自动调用这个采样器，实现打乱数据。这里使用另外一个很有用的采样方法：**WeightedRandomSampler**，它会根据每个样本的权重选取数据，在样本比例不均衡的问题中，可用它来进行重采样。 replacement 用于指定是否可以重复选取某一个样本，默认为True，即允许在一个epoch中重复采样某一个数据。

### 3. 定义collate_fn：

```
def detection_collate(batch):
    """Custom collate fn for dealing with batches of images that have a d
    number of associated object annotations (bounding boxes).

    Arguments:
        batch: (tuple) A tuple of tensor images and lists of annotations

    Return:
        A tuple containing:
            1) (tensor) batch of images stacked on their 0 dim
            2) (list of tensors) annotations for a given image are stacke
                                 0 dim
    """
    targets = []
    imgs = []
    for sample in batch:
```

```
        imgs.append(sample[0])
        targets.append(torch.FloatTensor(sample[1]))
    return torch.stack(imgs, 0), targets
```

使用dataloader时加入collate_fn参数，即可合并样本列表以形成小批量的Tensor对象，如果你的标签不止一个的话，还可以支持自定义，在上述方法中再额外添加对应的label即可。

```
data_loader = torch.utils.data.DataLoader(dataset, args.batch_size,
    num_workers=args.num_workers, sampler=sampler, shuffle=False,
    collate_fn=detection_collate, pin_memory=True, drop_last=True)
```

**参考链接：**

torch.utils.data - PyTorch master documentationpytorch.org（https://pytorch.org/docs/stable/data.html?highlight=dataloader#torch.utils.data.DataLoader）

Guidelines for assigning num_workers to DataLoaderdiscuss.pytorch.org（https://discuss.pytorch.org/t/guidelines-for-assigning-num-workers-to-dataloader/813）

**最新CV竞赛：** 2021 高通人工智能应用创新大赛 | CVPR 2021 | Short-video Face Parsing Challenge | 3D人体目标检测与行为分析竞赛开赛，奖池7万+，数据集达16671张！



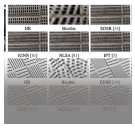
**觉得有用麻烦给个在看啦~**

阅读原文

喜欢此内容的人还喜欢

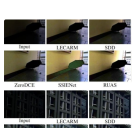
ICCV 2023 | 南开程明明团队提出适用于SR任务的新颖注意力机制（已开源）

极市平台



ICCV23 | 将隐式神经表征用于低光增强，北大张健团队提出NeRCo

极市平台



## ICCV 2023 | Pixel-based MIM: 简单高效的多级特征融合自监督方法

极市平台