


PyTorch之torch.utils.data.DataLoader详解

 进击的程小白  于 2021-07-19 21:20:37 发布  1865  收藏 15 版权

分类专栏: [Pytorch](#) 文章标签: [python](#) [计算机视觉](#)

 Pytorch 专栏收录该内容

0 订阅 1 篇文章 [订阅专栏](#)

ASS

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

[SOURCE]

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See [torch.utils.data](#) documentation page for more details.

dataset: (数据类型 dataset)

的数据类型,这里是原始数据的输入。PyTorch 内也有这种数据结构。

batch_size: (数据类型 int)

训练数据量的大小,根据具体情况设置即可(默认: 1)。PyTorch训练模型时调用数据不是一行一行进行(这样太没效率),而是一捆一捆来的。这里就是定义每次喂给神经网络多少行数据,如果设置成1,那就是一行一行进行(个人偏好,PyTorch默认设置是1)。每次是随机读取大小为batch_size。如果dataset中数据个数不是batch_size的整数倍,这最后一次把剩余的数据全部输出。若想把剩下的不足batch_size个数据丢弃,则将drop_last设置为True,会将多出来不足一个batch的数据丢弃。

shuffle: (数据类型 bool)

默认设置为False。在每次迭代训练时是否将数据洗牌,默认设置是False。将输入数据的顺序打乱,为了使数据更具有独立性,但如果数据是有序列特征的,就不要设置成True了。

collate_fn: (数据类型 callable, 没见过的类型)

一小段数据合并成数据列表,默认设置是False。如果设置成True,系统会在返回前会将张量数据(tensors)复制到CUDA内存中。

batch_sampler: (数据类型 Sampler)

采样,默认设置为None。但每次返回的是一批数据的索引(注意:不是数据)。其和batch_size、shuffle、sampler and drop_last参数是不兼容的。我想,应该是每次输入网络的数据是随机采样模式,才能使数据更具有独立性质。所以,它和一捆一捆按顺序输入,数据洗牌,数据采样,等模式是不兼容

sampler: (数据类型 Sampler)

默认设置为None。根据定义的策略从数据集中采样输入。如果定义采样规则,则洗牌(shuffle)必须为False。

num_workers: (数据类型 Int)

工作者数量,默认是0。使用多少个子进程来导入数据。设置为0,就是使用主进程来导入数据。注意:这个必须是大于等于0的,负数估计会出错。

pin_memory: (数据类型 bool)

 进击的程小白 [关注](#)



缓存，默认为False。在数据返回前，是否将数据复制到CUDA内存中。

drop_last: (数据类型 bool)

最后数据，默认为False。设置了 batch_size 的数目后，最后一批数据未必是设置的数目，有可能会小这时你是否需要丢弃这批数据。

timeout: (数据类型 numeric)

，默认为0。是用来设置数据读取的超时时间的，但超过这个时间还没读取到数据的话就会报错。所数值必须大于等于0。

worker_init_fn (数据类型 callable, 没见过的类型)

程导入模式，默认为Noun。在数据导入前和步长结束后，根据工作子进程的ID逐个按顺序导入数据。

atch_size举例分析:

```
1 """
2     批训练，把数据变成一小批一小批数据进行训练。
3     DataLoader就是用来包装所使用的数据，每次抛出一批数据
4 """
5 import torch
6 import torch.utils.data as Data
7
8 BATCH_SIZE = 5
9
10 x = torch.linspace(1, 11, 11)
11 y = torch.linspace(11, 1, 11)
12 print(x)
13 print(y)
14 # 把数据放在数据库中
15 torch_dataset = Data.TensorDataset(x, y)
16 loader = Data.DataLoader(
17     # 从数据库中每次抽出batch size个样本
18     dataset=torch_dataset,
19     batch_size=BATCH_SIZE,
20     shuffle=True,
21     # num_workers=2,
22 )
23
24 def show_batch():
25     for epoch in range(3):
26         for step, (batch_x, batch_y) in enumerate(loader):
27             # training
28             print("steop:{}, batch_x:{}, batch_y:{}".format(step, batch_x, batch_y))
29
30 if __name__ == '__main__':
31     show_batch()
```

输出为:

```
1 tensor([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11.])
2 tensor([11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  1.])
3
4 steop:0, batch_x:tensor([ 3.,  2.,  8., 11.,  1.]), batch_y:tensor([ 9., 10.,  4.,  1., 11.])
5 steop:1, batch_x:tensor([ 5.,  6.,  7.,  4., 10.]), batch_y:tensor([7.,  6.,  5.,  8.,  2.])
6 steop:2, batch_x:tensor([9.]), batch_y:tensor([3.]) 6
7 steop:0, batch_x:tensor([ 9.,  7., 10.,  2.,  4.]), batch_y:tensor([ 3.,  5.,  2., 10.,  8.])
8 steop:1, batch_x:tensor([ 5., 11.,  3.,  6.,  8.]), batch_y:tensor([7.,  1.,  9.,  6.,  4.])
9 steop:2, batch_x:tensor([1.]), batch_y:tensor([11.]) 9
10 steop:0, batch_x:tensor([10.,  5.,  7.,  4.,  2.]), batch_y:tensor([ 2.,  7.,  5.,  8., 10.])
11 steop:1, batch_x:tensor([3.,  9.,  1.,  8.,  6.]), batch_y:tensor([ 9.,  3., 11.,  4.,  6.])
12 steop:2, batch_x:tensor([11.]), batch_y:tensor([1.])12
13 Process finished with exit code 0
```

drop_last=True



进击的程小白

关注



```
1 """
2     批训练，把数据变成一小批一小批数据进行训练。
3     DataLoader就是用来包装所使用的数据，每次抛出一批数据
4 """
5 import torch
6 import torch.utils.data as Data
7
8 BATCH_SIZE = 5
9
10 x = torch.linspace(1, 11, 11)
```



的输出为：

```
1
2 tensor([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11.])
3 tensor([11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  1.])
4
steop:0, batch_x:tensor([ 9.,  2.,  7.,  4., 11.]), batch_y:tensor([ 3., 10.,  5.,  8.,  1.])
5
steop:1, batch_x:tensor([ 3.,  5., 10.,  1.,  8.]), batch_y:tensor([ 9.,  7.,  2., 11.,  4.])
6
steop:0, batch_x:tensor([ 5., 11.,  6.,  1.,  2.]), batch_y:tensor([ 7.,  1.,  6., 11., 10.])
7 steop:1, batch_x:tensor([ 3.,  4., 10.,  8.,  9.]), batch_y:tensor([9.,  8.,  2.,  4.,  3.])
8 steop:0, batch_x:tensor([10.,  4.,  9.,  8.,  7.]), batch_y:tensor([2.,  8.,  3.,  4.,  5.])
9
steop:1, batch_x:tensor([ 6.,  1., 11.,  2.,  5.]), batch_y:tensor([ 6., 11.,  1., 10.,  7.])
10
11 | Process finished with exit code 0
```

文章知识点与官方知识档案匹配，可进一步学习相关知识

thon入门技能树 人工智能 深度学习 32786 人正在系统学习中



创作挑战赛

新人创作奖励来咯，坚持创作打卡瓜分现金大奖

>

推荐

- [python torch.utils.data.DataLoader使用方法](#)09-17

介绍了python torch.utils.data.DataLoader使用方法，文中通过示例代码介绍的非常详细，对大家的学习或者工作具...
- [pytorch源码分析之torch.utils.data.Dataset类和torch.utils.data.DataLoader类](#)Never-Giveup的博客 3万+

之前介绍Pytorch深度学习框架优势之一是python优先，源代码由python代码层和C语言代码层组成，一般只需要理...
- [orch.utils.data.DataLoader\(\)详解_skywf的博客-CSDN...](#)3-3

中文文档里面没有写这个类但是我们经常用它,所以这里进行一下分析官网链接类定义参数额外信息使用方法以及要...
- [PyTorch入门学习:torch.utils.data.DataLoader_山上有...](#)3-19

orch中数据读取的一个重要接口是torch.utils.data.DataLoader。只要是用PyTorch来训练模型基本都会用到该接口,该...
- [orch.utils.data.DataLoader\(\)的使用](#)Fighting Hua 247


加载器，结合了数据集和取样器，并且可以提供多个线程处理数据集。在训练模型时使用到此函数，用来把训练数据...
- [pytorch技巧 五：自定义数据集 torch.utils.data.DataLoader 及Dataset的使用](#)qq_40788447的博客 2943

rch技巧 五：自定义数据集 torch.utils.data.DataLoader 及Dataset的使用 本博客中有可直接运行的例子，便于直观...
- [【PyTorch】torch.utils.data.DataLoader 简单介绍与使...](#)4-9

目录一、 torch.utils.data.DataLoader 简介二、实例参考链接一、 torch.utils.data.DataLoader 简介作用:torch.utils.dat...

[orch.utils.data.DataLoader函数详解_依旧seven的博客...](#)

s DataLoader(object): r""" Data loader. Combines a dataset and a sample



进击的程小白

关注