深入剖析DataLoder源码

2022.07.20

本文是在官方文档基础上深层剖析DataLoder，主要对sampler、batch_sampler、RandomSampler与SequentialSampler、collate_fn进行介绍

1.sampler与batch_sampler

```
if sampler is not None and shuffle:
    raise ValueError('sampler option is mutually exclusive with '
                     'shuffle')

if batch_sampler is not None:
    # auto_collation with custom batch_sampler
    if batch_size != 1 or shuffle or sampler is not None or drop_last:
        raise ValueError('batch_sampler option is mutually exclusive '
                         'with batch_size, shuffle, sampler, and '
                         'drop_last')
```

总的来说，sampler的作用就是以某种顺序从DataSet中取样本，batch_sampler的作用就是把sampler中的元素拼成一个个batch，返回的不是样本，而是样本对应的索引（以下会通过源码进行解释说明）

- 如果设置了sampler就不需要再设置shuffle
- 如果设置了batch_sampler就不需要再设置batch_size、shuffle、sampler、drop_last

2.RandomSampler与SequentialSampler

```
if sampler is None:  # give default samplers
    if self._dataset_kind == _DatasetKind.Iterable:
        # See NOTE [ Custom Samplers and IterableDataset ]
        sampler = _InfiniteConstantSampler()
    else:  # map-style
        if shuffle:
            sampler = RandomSampler(dataset, generator=generator)  # type:
ignore[arg-type]
        else:
            sampler = SequentialSampler(dataset)  # type: ignore[arg-type]
```

- **RandomSampler源码：**

```
class SubsetRandomSampler(Sampler[int]):
    r"""Samples elements randomly from a given list of indices, without
replacement.

    Args:
        indices (sequence): a sequence of indices
        generator (Generator): Generator used in sampling.
    """
    indices: Sequence[int]
    def __init__(self, indices: Sequence[int], generator=None) -> None:
        self.indices = indices
        self.generator = generator

    def __iter__(self) -> Iterator[int]:
        for i in torch.randperm(len(self.indices), generator=self.generator):
            yield self.indices[i]

    def __len__(self) -> int:
        return len(self.indices)
```

当没有设置Sampler时，如果设置了shuffle，那么将调用RandomSampler：返回的是一个**随机**的索引范围为[0,n-1]的列表（关键是torch.randperm方法）

- **SequentialSampler源码：**

```
class SequentialSampler(Sampler[int]):
    r"""Samples elements sequentially, always in the same order.

    Args:
        data_source (Dataset): dataset to sample from
    """
    data_source: Sized

    def __init__(self, data_source: Sized) -> None:
        self.data_source = data_source

    def __iter__(self) -> Iterator[int]:
        return iter(range(len(self.data_source)))

    def __len__(self) -> int:
        return len(self.data_source)
```

如果设置了shuffle，那么将调用SequentialSampler：返回的是索引列表[0,1,....,n-1]，是按原始顺序的，没有打乱（关键是range函数）

3.batch_sampler源码（iter函数）

```
def __iter__(self) -> Iterator[List[int]]:
    # Implemented based on the benchmarking in
https://github.com/pytorch/pytorch/pull/76951
    if self.drop_last:
        sampler_iter = iter(self.sampler)
        while True:
            try:
                batch = [next(sampler_iter) for _ in range(self.batch_size)]
                yield batch
            except StopIteration:
                break
    else:
        batch = [0] * self.batch_size
        idx_in_batch = 0
        for idx in self.sampler:
            batch[idx_in_batch] = idx
            idx_in_batch += 1
            if idx_in_batch == self.batch_size:
                yield batch
                idx_in_batch = 0
                batch = [0] * self.batch_size
        if idx_in_batch > 0:
            yield batch[:idx_in_batch]
```

iter函数的作用是batch_sampler如何从sampler中取样本索引，drop_last是指是否丢弃最后的一些元素，这些元素可能构不成一个batch

- 如果设置了drop_last，先设置一个sampler_iter，表明可以从这个迭代器中通过next方法取元素，然后再通过循环从该迭代器中取batch_size大小的索引数目，然后返回该索引列表即可（到达batch_size大小即返回，说明设置了drop_last）
- 如果没有设置drop_last，可以看到前面语句返回的都是batch_size大小的索引列表，最后一个if，返回的是batch[:idx_in_batch]，说明最后一些没有返回，被丢弃了

4.collate_fn

如果自己想写一个collate_fn，那么就以batch作为输入，比如想对feature做一个padding，那么就先从batch中取出feature，然后把一个batch中feature的最大长度计算出来，把其余的feature都padding成这个长度即可

5.总结

总结DataSet与DataLoder的使用方法：

1. 按照pytorch官网自定义DataSet的方法定义自己的DataSet，主要是对单个样本，其中可以自定义transfrom方法对数据和标签进行处理，最终返回的如 image,label 这样的样本对
2. 得到DataSet后，把它们放入DataLoder中，拼成一个个mini_batch，其中涉及到sampler、batch_sampler、collate_fn，根据自己需要来设计