

# pytorch里面nn.Module讲解

原创

这题要怎么做 于 2020-03-30 22:39:03 发布 2534 收藏 16

版权

分类专栏: pytorch

机器学习



pytorch 同时被 2 个专栏收录

2 订阅

8 篇文章

订阅专栏

`nn.Module` 是在 `pytorch` 使用非常广泛的类，搭建网络基本都需要用到这个。

当我们搭建自己的网络时，可以继承官方写好的 `nn.Module` 模块，为什么要用这个呢？好处如下：

## nn.Module作用

1. 可以提供一些现成的基本模块比如：
2. 容器
3. 参数管理
4. 所有modules的节点 孩子节点都是直系的
5. `to(device)`
6. 保存和加载模型
7. 训练/测试
8. 实现自己的类

### 8.1 举一个自己写的线性层的例子

## 1. 可以提供一些现成的基本模块比如：

`Linear`、`ReLU`、`Sigmoid`、`Conv2d`、`Dropout`

不用自己一个一个的写这些函数了，这也是为什么我们用 `框架` 的原因之一吧。

## 2. 容器

比如我们经常用到的 `nn.Sequential()`，顾名思义，将网络模块封装在一个容器中，可以方便网络搭建

如下面一个例子：

```
1 class TestNet(nn.Module):
2     def __init__(self):
3         super(TestNet, self).__init__()
4         self.net = nn.Sequential(nn.Conv2d(1, 16, stride=1, padding=1),
5                                   nn.MaxPool2d(2, 2),
6                                   Flatten(),
7                                   nn.Linear(1*14*14, 10))
8     def forward(self, x):
9         return self.net(x)
```

## 3. 参数管理

参数名字可以自动生成（想想如果自己去命名，百万参数的网络没法搭建），然后这些参数都可以传到优化器里面去优化

## 4. 所有modules的节点 孩子节点都是直系的

```
1 class BasicNet(nn.Module):
2     def __init__(self):
```



这题要怎么做

关注



```
3         super(BasicNet, self).__init__()
4         self.net = nn.Linear(4, 3)
5
6     def forward(self, x):
7         return self.net(x)
8
9 class Net(nn.Module):
10     def __init__(self):
11         super(Net, self).__init__()
12         self.net = nn.Sequential(BasicNet(),
13                                   nn.ReLU(),
14                                   nn.Linear(3, 2))
15     def forward(self, x):
16         return self.net(x)
```

比如上面的代码，我们可以看出Net网络中有5个孩子节点：`nn.Sequential`，`BasicNet`，`nn.ReLU`，`nn.Linear`，`BasicNet`里面的`nn.Linear`

## 5.to(device)

`nn.Module` 还有一个功能是将某个网络所有成员、函数、操作都搬移到GPU上面。采用代码如下：

```
1 device = torch.device('cuda')
2 net = Net()
3 net.to(device)
```

上面device代表当前的设备是GPU还是CPU，需要注意的是为什么我们不写

```
1 net = net.to(device)
```

其实效果是一样的，采用 `nn.Module` 模块，net加上 `.to(device)`，还是net。如果是变量则不是一样的，即如果对于tensor `bias`，那么 `bias` 和 `bias.to(device)` 不是一样的，则需要重新命名。

## 6.保存和加载模型

可以方面我们保存和加载模型

加载模型：

```
1 net.load_state_dict(torch.load('ckpt.mdl'))
```

保存模型：

```
1 torch.save(net.state_dict(), 'ckpt.mdl')
```

## 7.训练/测试

方便训练和测试进行切换，为什么？因为网络中Dropout和BN在训练和测试是不一样的，需要切换

如果不切换效果就会很差，**这个是容易犯的一个错误。**

```
1 net.train()
2 net.eval()
```

## 8.实现自己的类

官方给的模块还是基础操作的，如果自己要搭建复杂的操作也容易实现，一个典型的例子就是可以自己设计一个新的损失函数。

下面给出将tensor压平的例子（`nn.Module` 没有这个操作）：

```
1 class Flatten(nn.Module):
2     def __init__(self):
3         super(Flatten, self).__init__()
```



这题要怎么做

关注



```

4
5     def forward(self, input):
6         return input.view(input.size(0), -1)
7
8 class TestNet(nn.Module):
9     def __init__(self):
10         super(TestNet, self).__init__()
11         self.net = nn.Sequential(nn.Conv2d(1, 16, stride=1, padding=1),
12                                   nn.MaxPool2d(2, 2),
13                                   Flatten(), #自己定义的
14                                   nn.Linear(1*14*14, 10))
15
16     def forward(self, x):
17         return self.net(x)

```

`Flatten` 压平的操作则是我们自己构建的类，可以方便后续BasicNet类使用，注意 `nn.Sequential` 里面必须是类。

且在上面例子中 `Flatten` 不需要接任何参数。

## 8.1举一个自己写的线性层的例子

```

1 class MyLinear(nn.Module):
2     def __init__(self, inp, outp):
3         super(MyLinear, self).__init__()
4         # requires_grad = True
5         self.w = nn.Parameter(torch.randn(outp, inp))
6         self.b = nn.Parameter(torch.randn(outp))
7
8     def forward(self, x):
9         x = x @ self.w.t() + self.b
10        return x

```

在上面自己写的线性层  $y = wx + b$ ，可以看出  $w$  和  $b$  必须要使用 `nn.Parameter` 这个模块。原因是只用加上了 `nn.Parameter` 后， $w$  和  $b$  才可以用优化器SGD等进行优化。

如果不写 `nn.Parameter` 那么则需要写 `requires_grad = True`，还要自己写优化器，就很麻烦。用了 `Parameter` 可以方便我们优化网络：

```

1 model = MyLinear.to(device)
2 optimizer = optim.Adam(model.parameters(), lr=1e-3)
3 # backprop
4 optimizer.zero_grad()
5 loss.backward()
6 optimizer.step()

```

### 文章知识点与官方知识档案匹配，可进一步学习相关知识

Python入门技能树 人工智能 深度学习 32801 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

## 相关推荐

浅析PyTorch中nn.Module的使用

12-25

`torch.nn.Modules` 相当于是对网络某种层的封装，包括网络结构以及网络参数和一些操作 `torch.nn.M...`

PyTorch 1.0 系列学习教程 (3) : nn module

gukedream的专栏 4127

PyTorch 1.0: nn module NN MODULEPyTorch:nnPyTorch:optimPyTorch:Custom nn ModulesPyTor...

nn.module类搭建简单神经网络\_abysswatcher1的博客

3-3

`nn.Module`是PyTorch提供的神经网络类,并在类中实现了网络各层的定义及前向计算与反向传播机制...

pytorch中nn.moudle模块\_热爱生活,热爱代码

`nn.moudle`是所有卷积神经网络的基类,相信一定非常困扰;



这题要怎么做

关注

