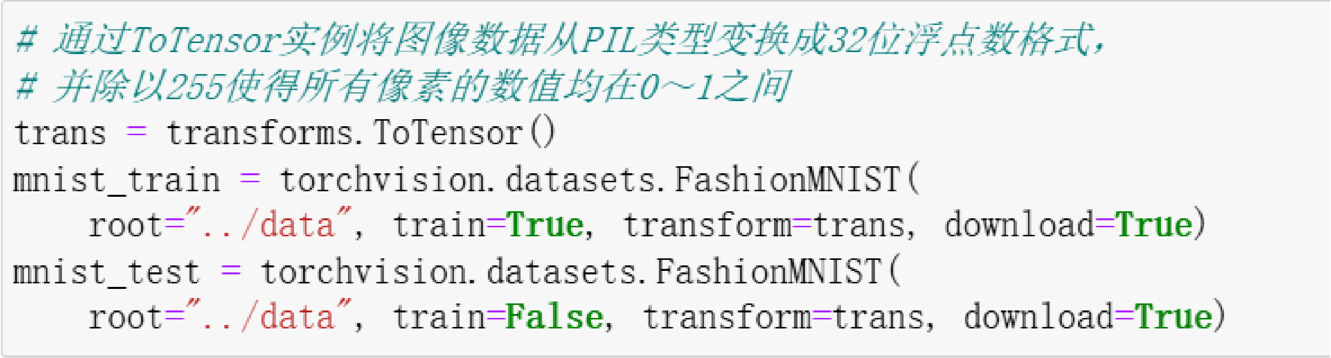
实现3.5图像分类的实验

实验目的：获取Fashion-Mnist数据集。

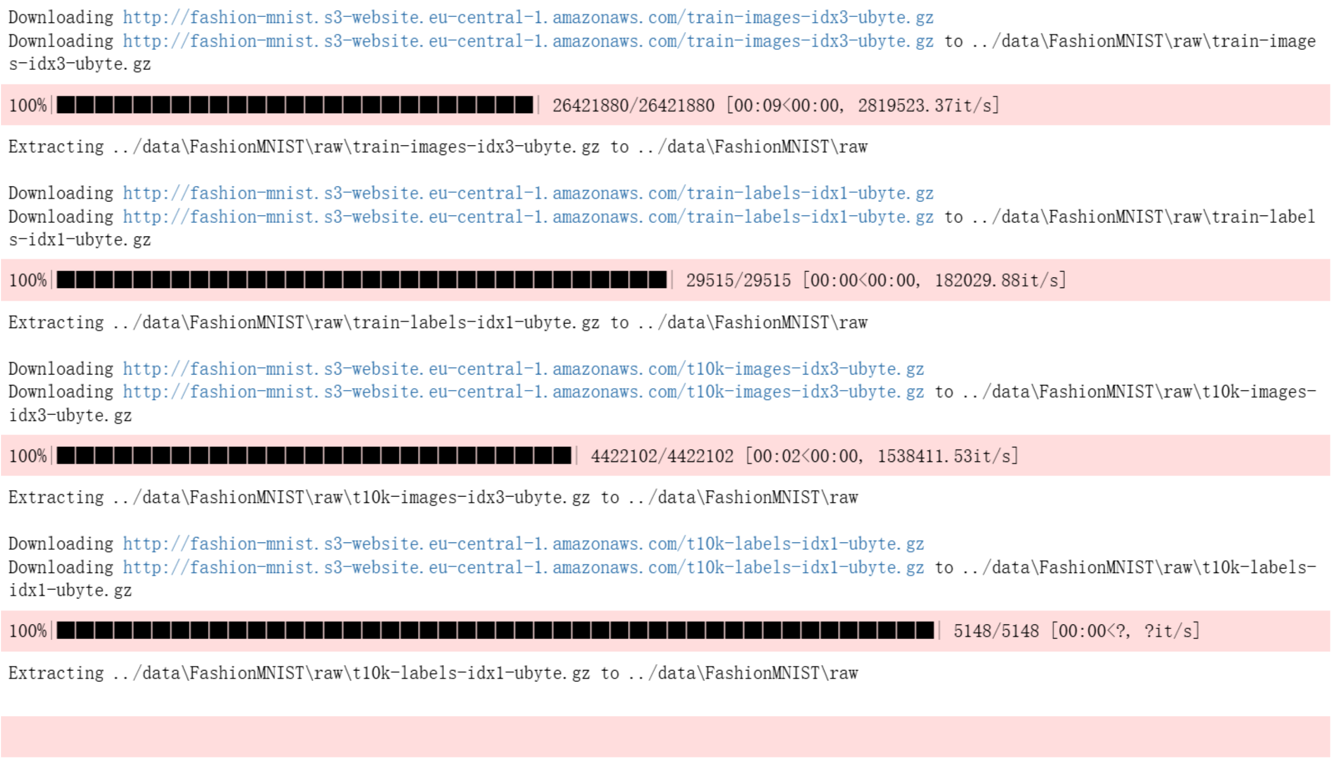
Fashion-Mnist数据集包含七万张灰度图像，其中有六万张训练集和一万张测试集。每张图像的大小为28×28。

读取数据集

Pytorch中的内置框架torchvision.datasets中含有服饰数据集，我们通过此函数下载数据集（如下图），并将图像数据转化为浮点数格式。



```
# 通过ToTensor实例将图像数据从PIL类型变换成32位浮点数格式，
# 并除以255使得所有像素的数值均在0～1之间
trans = transforms.ToTensor()
mnist_train = torchvision.datasets.FashionMNIST(
    root="../data", train=True, transform=trans, download=True)
mnist_test = torchvision.datasets.FashionMNIST(
    root="../data", train=False, transform=trans, download=True)
```

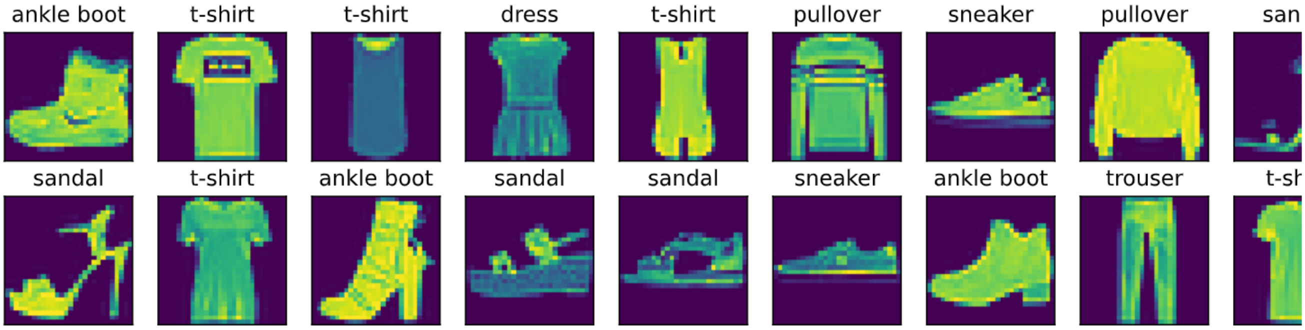


```
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-images-idx3-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-images-idx3-ubyte.gz to ../data\FashionMNIST\raw\train-image
s-idx3-ubyte.gz

100%|█████████████████████████| 26421880/26421880 [00:09<00:00, 2819523.37it/s]

Extracting ../data\FashionMNIST\raw\train-images-idx3-ubyte.gz to ../data\FashionMNIST\raw

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-labels-idx1-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-labels-idx1-ubyte.gz to ../data\FashionMNIST\raw\train-label
s-idx1-ubyte.gz

100%|████████████████████████████████| 29515/29515 [00:00<00:00, 182029.88it/s]

Extracting ../data\FashionMNIST\raw\train-labels-idx1-ubyte.gz to ../data\FashionMNIST\raw

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-images-idx3-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-images-idx3-ubyte.gz to ../data\FashionMNIST\raw\t10k-images-
idx3-ubyte.gz

100%|███████████████████████████| 4422102/4422102 [00:02<00:00, 1538411.53it/s]

Extracting ../data\FashionMNIST\raw\t10k-images-idx3-ubyte.gz to ../data\FashionMNIST\raw

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz to ../data\FashionMNIST\raw\t10k-labels-
idx1-ubyte.gz

100%|██████████████████████████████████████████████| 5148/5148 [00:00<?, ?it/s]

Extracting ../data\FashionMNIST\raw\t10k-labels-idx1-ubyte.gz to ../data\FashionMNIST\raw
```

展示

并展示前十八张图片（如下图）。

```
X, y = next(iter(data.DataLoader(mnist_train, batch_size=18)))
show_images(X.reshape(18, 28, 28), 2, 9, titles=get_fashion_mnist_labels(y));
```



实现3.7softmax回归实验

算法原理

一般的softmax实现

1. 对于特征集合X，将其经过线性组合得到预测的模型：

$$f(x;w) = w_1x_1 + w_2x_2 ... w_nx_n + b = w^Tx + b$$

2. 为了保证在任何数据上的输出都是非负的，使用softmax激励模型精确地估计概率：

$$\hat{y} = softmax(o) \text{ 其中 } \hat{y}_j = \frac{exp(o_j)}{\sum_k exp(o_k)}$$

3. 故softmax回归的矢量表达式的计算式为：

$$\begin{cases} O = XW + b \\ \hat{Y} = softmax(O) \end{cases}$$

4. 定义损失函数：在softmax回归中使用交叉熵损失函数。

$$H[P] = \sum_j -P(j)logP(j)$$

简洁版的softmax实现

相比于以上的例子，对softmax激励模型进行了修改。

为了避免数值非常大的值影响结果，出现上溢问题。在计算softmax之前，首先减去一个$max(o_k)$。

$$\hat{y}_j = \frac{exp(o_j - max(o_k))exp(max(o_k))}{\sum_k exp(o_k - max(o_k))exp(max(o_k))}$$

代码实现与结果

一般的softmax实现

1. 定义softmax操作

```python
def softmax(x):
    X_exp = torch.exp(x)
    partition = X_exp.sum(1, keepdim=True)
    return X_exp / partition
```

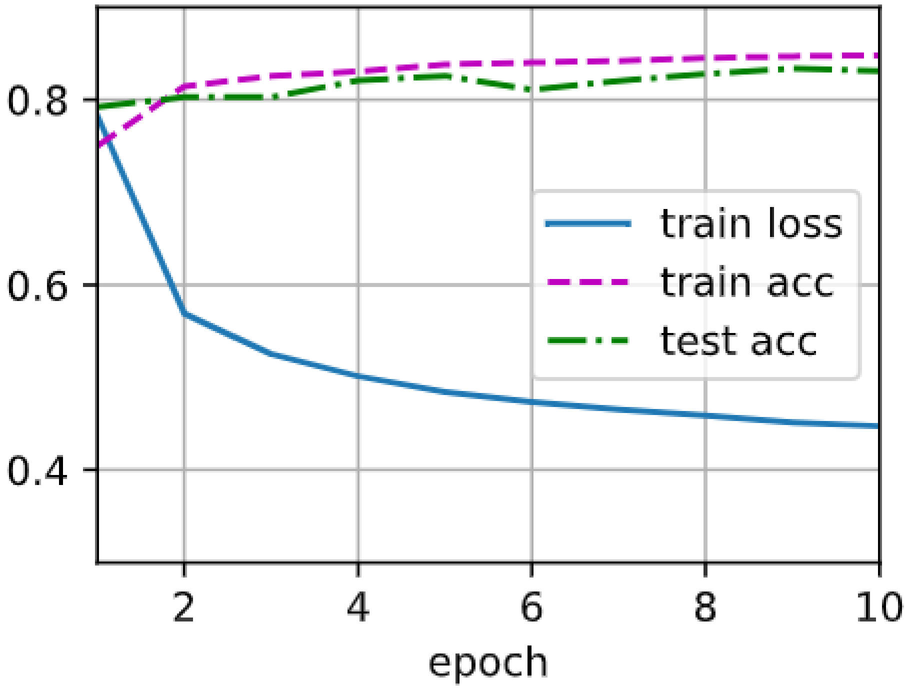
2. 定义模型

```python
def net(X):
    return softmax(torch.matmul(X.reshape((-1, W.shape[0])), W) + b)
```

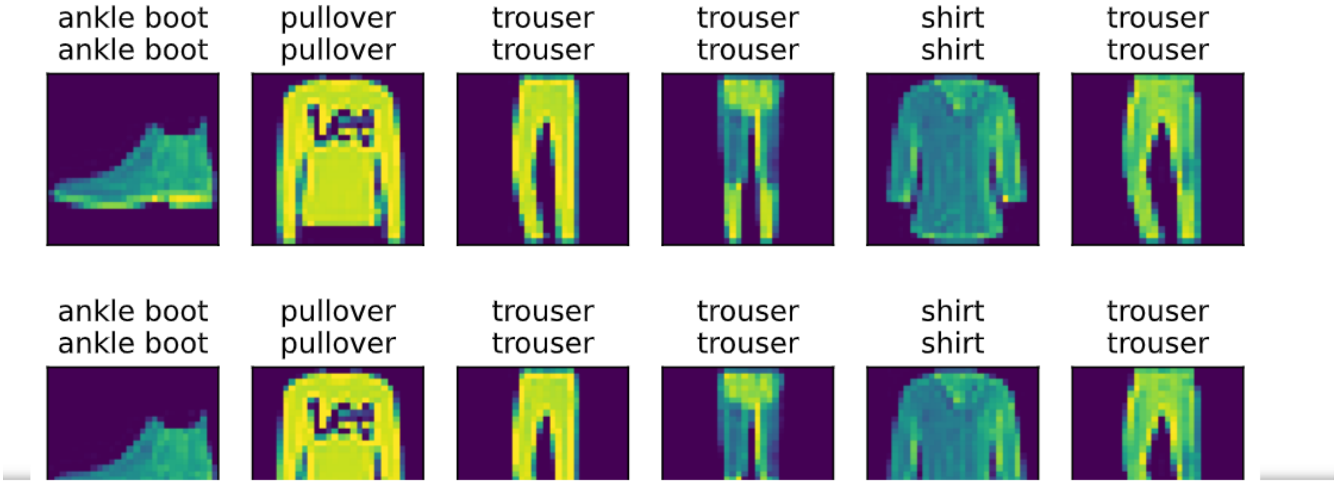
3. 定义损失函数

```python
def cross_entropy(y_hat, y):
    return - torch.log(y_hat[range(len(y_hat)), y])

cross_entropy(y_hat,y)
```

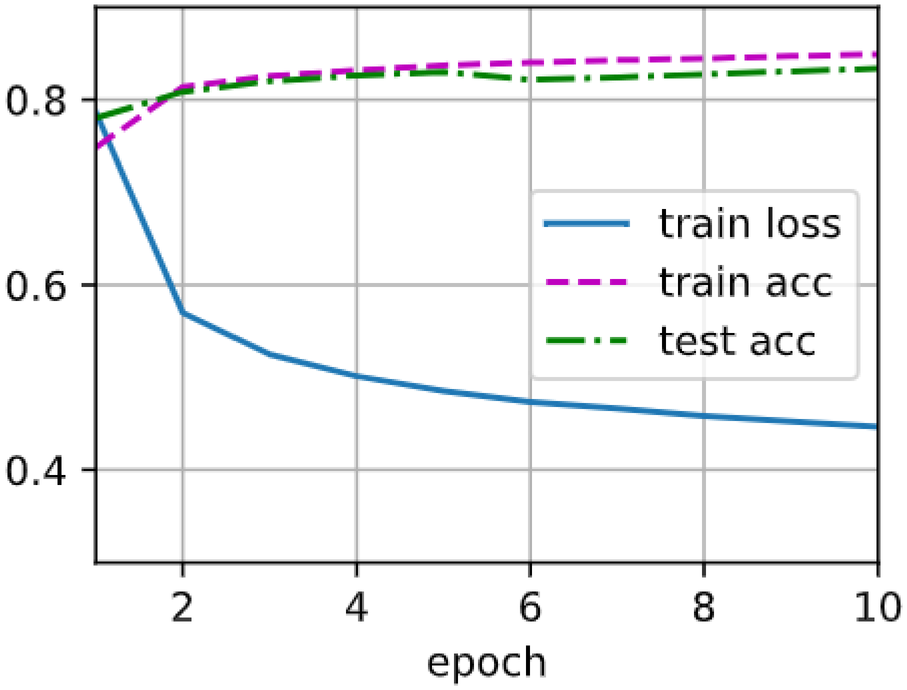
4. 模型训练



5. 模型预测



简洁版的softmax实现


epoch
train loss
train acc
test acc