作业 3 Python 人工智能应用

许竞帆 2019010406

1 图片可视化

图 1

2 训练模型并推理

图 2

对学号最后一位预测正确。

3 绘制 loss 曲线

使用 matplotlib，对每个 epoch 的所有 step 的 loss 求和作图如下：

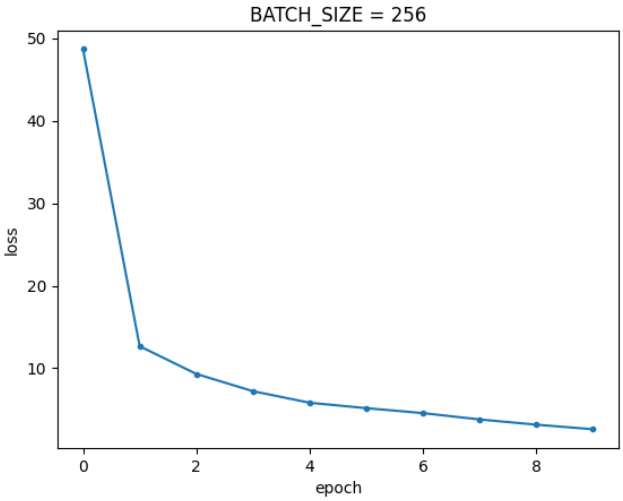

图 3

4 更换优化器

各参数不变，简单更改即可。

```
optimizer = torch.optim.SGD(model.parameters(), lr=learning_rate)
```

5 添加数据预处理

使用 torchvision 包对 data_augment_transform 进行修改，根据 loss 曲线评价模型训练效果。

5.1 随机剪裁

调用 RandomResizedCrop，随机裁剪图片并重新放大至 28*28，再进行训练。

```
def data_augment_transform():
    data_augment = torchvision.transforms.Compose([
        torchvision.transforms.RandomResizedCrop(28),
        torchvision.transforms.ToTensor(),
    ])
    return data_augment
```

正常情况下，进行随机裁剪的数据集训练效果一般会更好。但是在本次作业中，相比未进行裁剪的训练集，模型精确度下降，loss 也更大。可能是图片中大部分为黑底背景，只有少部分是数字的组成，随机裁剪后可能没有特征，训练效果不好。

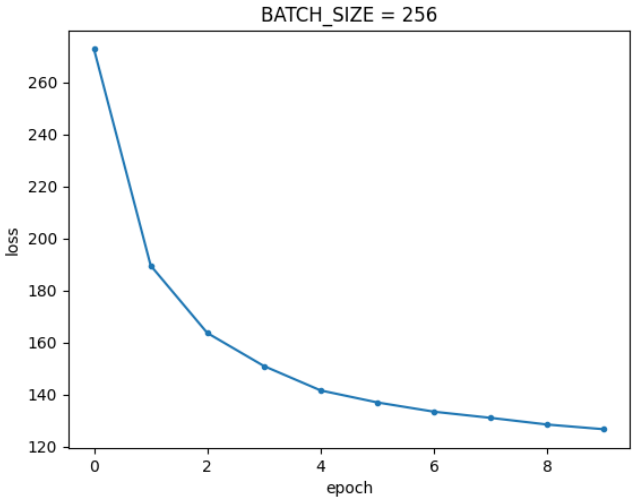

图 4

5.2 水平翻转

调用 RandomHorizontalFlip，以 0.5 的概率随机水平翻转图片。

```
def data_augment_transform():
    data_augment = torchvision.transforms.Compose([
        torchvision.transforms.RandomHorizontalFlip(),
        torchvision.transforms.ToTensor(),
    ])
    return data_augment
```

模型精确度比未翻转时较小，loss 之和变大，但相比于随机裁剪的训练模型，水平翻转的模型与未翻转时的 loss 之和相差更小。可能是因为其中一部分被翻转了，而另一部分没有被翻转，因此差别不大。

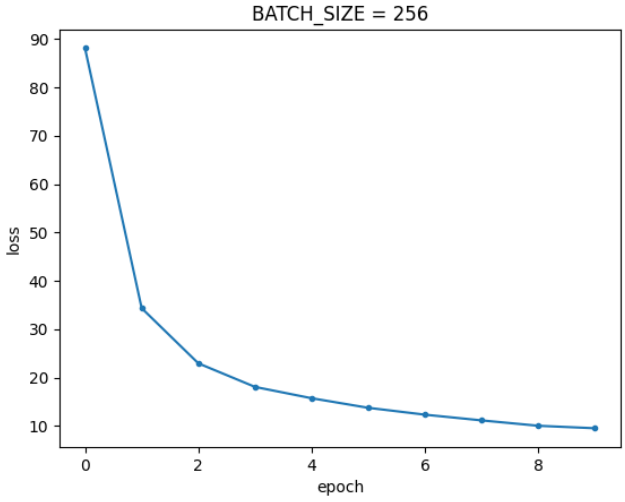

图 5

5.3 垂直翻转

调用 RandomHorizontalFlip，以 0.5 的概率随机垂直翻转图片。

```
def data_augment_transform():
    data_augment = torchvision.transforms.Compose([
        torchvision.transforms.RandomVerticalFlip(),
        torchvision.transforms.ToTensor(),
    ])
    return data_augment
```

结果与水平翻转相似，可能的原因也应和水平翻转相同。

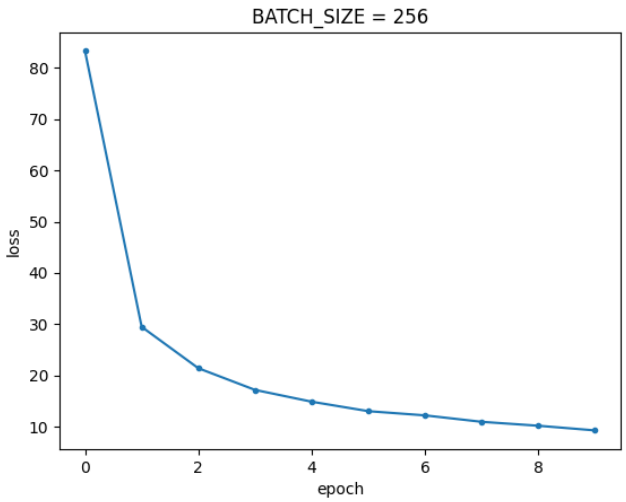

图 6

6 模型导出及模型可视化

使用 https://lutzroeder.github.io/netron/可视化工具，对添加一层后的卷积网络进行可视化如下。

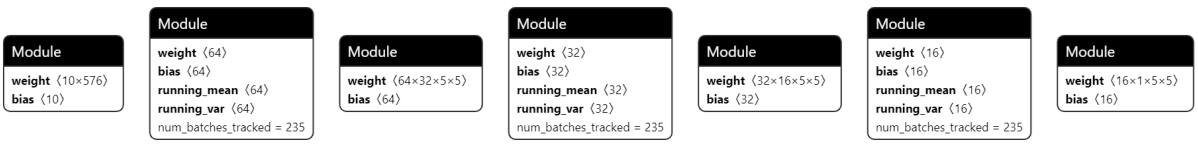

图 7

7 错误样例分析

输出其中 5 个模型推理错误如下，图片下表中字母 P 后数字代表模型预测结果，L 后数字代表标签。模型推理错误的可能原因是这些推理错误的图片中数字并不规则，具有在具有较多的预测出来的数字的特征，例如（a）中实际为 5，推理为 0，（b）中实际为 6，推理为 0，可能是

因为图片中存在一个较大的圆，而其他特征较不明显，与 0 的特征更相符。对于（c）和（d）都被推理为 2，可能是因为图中有较为明显的 2 的折叠的特征。对于（e）中数字的错误推理，可能是 7 和 9 的主要差别在于起笔的一横和一个圆，而图中弧度较大，被推理为 9，也有可能是 7 和 9 本身较难区分，因为存在另一种中间还有一横的手写体数字 7。

(a) P0L5 (b) P0L6 (c) P2L6 (d) P2L7 (e) P9L7

图 8