华中科技大学

基于 CNN 的 CIFAR-10 数据集分类

专　业：　计算机科学与技术

班　级：　CS2005 班

学　号：　U202090063

姓　名：　董玲晶

指导教师：　刘 康

分数	
教师签名	

2023 年 4 月 1 日

1 任务要求

设计一个卷积神经网络，在 CIFAR-10 数据集上实现分类任务。不能直接导入现有的 CNN 网络，比如 VGG、ResNet 等，可以以现有网络为基础进行改进，深度学习框架任选。

可以尝试不同的卷积神经网络设计、使用不同的激活函数等，观察网络性能。实验报告包含网络设计、在 10 类测试集上的平均准确率截图、每一类的准确率截图以及必要的分析等。

2 任务设计

2.1 数据集处理

使用 torchvision,datasets 导入 CIFAR10 的训练集和测试集，并对数据集做变换预处理。定义如下 transform_train 和 transform_test，从 transforms 模块导入 transforms 函数对输入的图片数据集进行一系列变换：对图像进行随机裁剪、随机水平翻转、转换为 PyTorch 张量并做标准化处理。做标准化处理时用到的参数为 CIFAR10 数据集的均值和标准差序列，此数据从网上获得。

导入数据集后使用 torch.utils.data.DataLoader 打乱并按批次打包，等待后续训练使用。

代码 1 CIFAR10 数据预处理

```
transform_train = transforms.Compose([
  transforms.RandomCrop(32, padding=4),
  transforms.RandomHorizontalFlip(),
  transforms.ToTensor(),
  transforms.Normalize((0.4914, 0.4822, 0.4465), (0.2023, 0.1994, 0.2010)),
])
transform_test = transforms.Compose([
  transforms.ToTensor(),
  transforms.Normalize((0.4914, 0.4822, 0.4465), (0.2023, 0.1994, 0.2010)),
])
```

2.2 网络结构

神经网络结构的确定分为以下三个阶段，而具体对应的网络性能和详情将在下一章节中进行讨论：

- 只使用几个卷积层激活并池化提取特征，再用全连接层分类输出；
- 对数据集进行预处理做数据增强，并添加 BatchNorm 层；
- 更改网络结构，添加残差块和增加通道数；

综合来说，本模型参考了 VGG 和 ResNet18 的一些网络结构特性，如构建残差块使用，每个卷积层后跟一个 ReLU 激活函数、池化层采用最大池化、小卷积核使用等。另外在卷积层和激活函数之间添加了批量归一化操作，用于对每个通道的标准化。

网络总体框架如下表所示（BATCH_SIZE 为 128）：

表 1 网络总体结构 ResNet

	代码	备注
0	nn.Conv2d(3, 32, 3)	32 个 3×3×3 的卷积核，padding=1
1	nn.BatchNorm2d(32)	32 个通道批量归一化
2	nn.ReLU( )	激活函数
3	nn.MaxPool2d(2, 2)	下采样最大池化，图片长宽变原来一半
插入第一个残差块：in_channels(32) => out_channels(64)		
4	nn.Conv2d(64, 128, 3)	128 个 64×3×3 的卷积核，padding=1
5	nn.BatchNorm2d(128)	64 个通道批量归一化
6	nn.ReLU( )	激活函数
7	nn.MaxPool2d(2, 2)	下采样最大池化，图片长宽变原来一半
插入第二个残差块：in_channels(128) => out_channels(256)		
8	nn.AdaptiveAvgPool2d((1, 1))	自适应平均池化，将图片长宽变为 1
9	nn.Linear(256, 10)	分类
10	nn.SoftMax(dim=1)	输出分类标量

表 2 残差块结构 ResidualBlock

	代码	备注
0	nn.Conv2d(in, out, 3)	out 个 in×3×3 的卷积核，padding=1
1	nn.BatchNorm2d(out)	out 个通道批量归一化
2	nn.ReLU( )	激活函数
3	nn.Conv2d(out, out, 1)	out 个 out×1×1 的卷积核，padding=1
4	nn.BatchNorm2d(out)	out 个通道批量归一化
5	out += down_sample(x)	将以上输出加上输入
备注：设输入为 x，每一层为 out		

2.3 架构及固定参数

总体采用 PyTorch 架构，训练、测试、绘图、网络等全体代码详情见开源仓库：https://github.com/Elubrazione/cv_labs_hust/tree/main/lab2

2.3.1 优化器

选用 torch.optim.Adam()，其是随机梯度下降（SGD）算法的一种变体，具有自适应学习率和动量参数。

2.3.2 损失函数

选用交叉熵损失函数 F.cross_entropy( )，其是一种用于比较两个概率分布之间差异的度量方式，通常用于多分类模型，输出通常是一个经过 Softmax 归一化后的概率分布，表示每个类别的预测概率。

2.3.3 其它参数

BATCH_SIZE = 128

MODEL = False/True，是否重载模型

3 实验结果与分析

本章节分为两大部分：第一部分为 2.2 中提到的不同网络结构对综合预测分类准确率的影响的讨论；第二部分则展示基于最终选定网络结构得到实验结果，如每一类的准确率和平均准确率，并对结果展开分析。

3.1 不同网络结构的性能

如 2.2 中所说，前后共使用了三种不同的网络结构。

使用简单的卷积层和全连接层迭代了 30 个 epoch 左右最终 10 类的平均准确率稳定在了 67%左右，无法突破更高。在前面网络的基础上增加了批量归一化层并对数据集做一些预处理后，准确率有了些许提高，达到了 70%，但显然效果仍欠佳。最后还是对网络结构进行了大改，即章节 2.2 中描述网络结构，在迭代了 25 次之后准确率达到 81%。这与最开始简单网络的 30 次才达到 67%相比，在预测准确率方面的提高程度是显著的。

卷积层数和通道数的增加带来的是模型复杂度和训练时间的增加，硬件的限制使得我在其它方面做了妥协，如在卷积层和全连接层之间添加的自适应池化和只使用一层全连接层，但从另外一个角度来看，这种操作也有些许利于提高模型的泛化能力、防止过拟合化的发生。

3.2 实验结果与分析

在最终确定了使用的网络结构后，一开始设置的初始学习率大小为 torch.optim.Adam 的默认值 0.001。在迭代了无数轮后，测试集上的损失率反复震荡，准确率卡在了 87%~88%左右。这时我改变了初始学习率（0.0009），重载了上一轮保存模型，在五次稳定在 88%以上后再次更改初始学习率（0.0005），此时准确率达到了 89%，详情如图 1 所示。

```
epoch241  acc 89.22
[902, 945, 876, 717, 900, 873, 914, 922, 937, 936]
epoch242  acc 89.03
[922, 943, 847, 779, 880, 862, 911, 908, 939, 912]
epoch243  acc 89.07000000000001
[895, 931, 868, 752, 896, 862, 909, 910, 939, 945]
epoch244  acc 89.16
[911, 938, 819, 798, 904, 857, 929, 886, 939, 935]
epoch245  acc 89.16
[905, 942, 825, 756, 905, 853, 927, 929, 946, 928]
```

图 1 lr=0.0005 迭代五轮的平均和每一类准确率

此后我上网查询，看到一个 tutorial 的实验证明，虽然 Adam 可以进行自适应的学习率调整，但进行学习率衰减处理仍然能提升模型的预测能力和泛化能力，使梯度下降地更加稳定并加速收敛、更快找到全局最优解。但运用此法后准确率在上升到 90.5%后陷入了平稳。图 2 展示了整个训练过程中预测分类的准确率随训练次数的变化，可以看到如之前所说，很长一段时间里（epoch75~240）准确率的变化不明显，改变了初始学习率后打破了瓶颈，准确率上升了一段后又几乎维持不变，且可以很明显地看出此段的震荡幅度远小于前面一段。

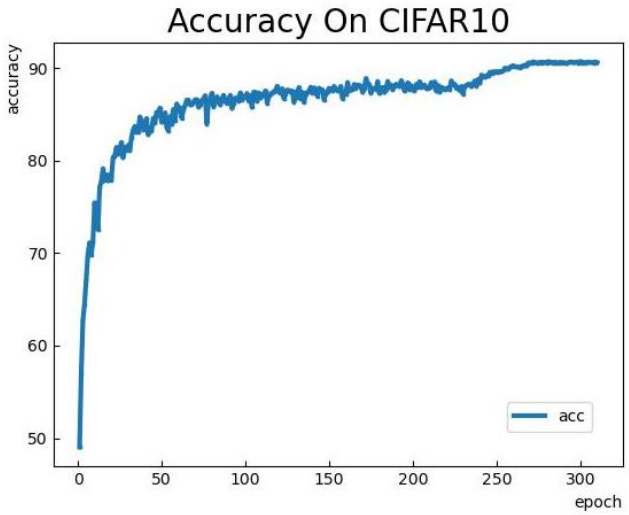

图 2 训练期间平均准确率的变化

最后的模型在测试集上的平均准确率为 90.58%，图 3 则显示了模型在测试集的每一类上的分类准确率。

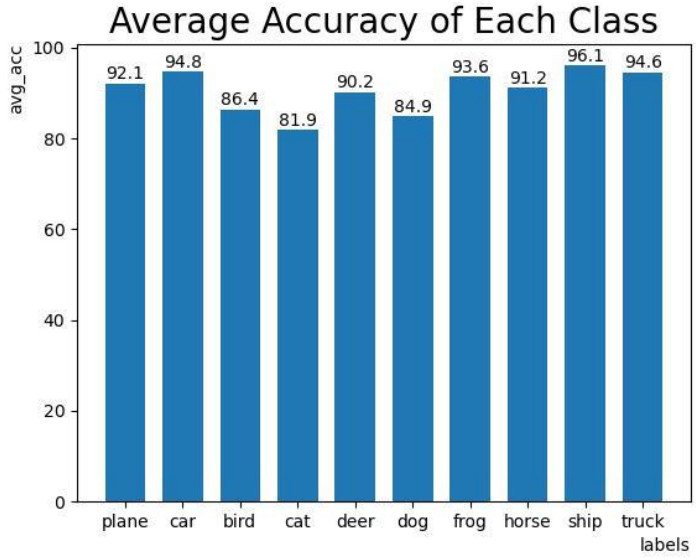

图 3 最终模型在测试集上的每一类预测准确率

从直方图 3 可以看出对猫、狗、鸟三类的预测准确率相较于其它类是偏低的，尤其是对猫的分类准确率只有 81.9%。其实我在训练的过程中通过观察每一类上的预测输出也发现了这个问题，哪怕在训练开始阶段，这三类分类正确的数目相对于其他类也较少，这里选取了 epoch36~40 的数据，绘成堆叠直方图如图 4 所示。

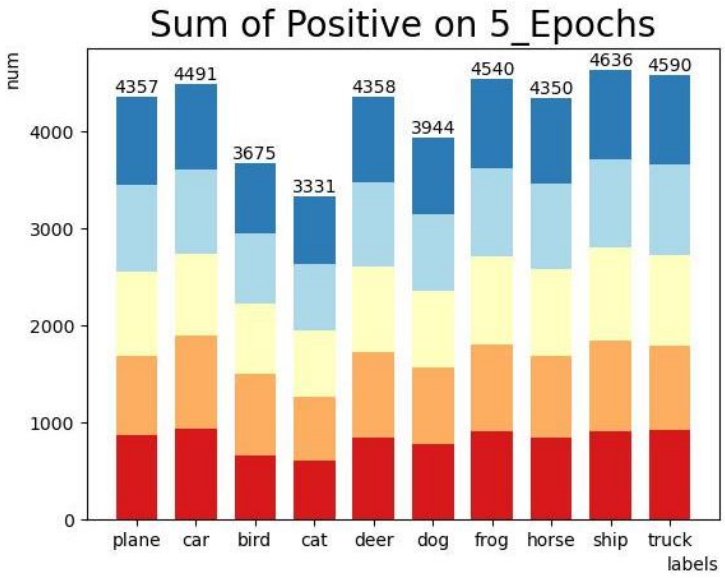

图 4 连续五次训练里每一类分类正确数目总和

这里推测这三类准确率较低的一个原因可能是由于猫、狗以及鸟的面部特征相对来说比较复杂，而且猫和狗的轮廓和外观较为相似。我所设计的神经网络结构虽然借鉴了 ResNet 的残差块和 VGG 用多个小卷积（3×3）核代替大卷积核的结构，按道理来说可以获得较好的性能，但卷积核大小为 3×3 的卷积层深度不够，造成特征提取能力的下降，最终导致了上图中的结果。

4 总结

本实验基于 CIFAR10 数据集，通过借鉴 VGG、ResNet 等经典神经网络结构，搭建了一个小规模 CNN 网络，并不断尝试调整参数对模型进行优化，最终在预测集上取得了平均准确率 90.58%的结果。此模型对具有复杂及相似特征的数据效果欠佳，或许可以考虑重构网络，增大卷积核的视野对模型进行进一步的改进。