学习使用常用 CNN 结构和数据增强技术*

本工程的持续时间为一周

November 19, 2020

1 目标

1. 了解并使用常用的 CNN 结构。随着 AlexNet 的出现，众多优秀的卷积网络结构已经被设计出来，例如 VGG, Inception, ResNet, ResNeXt, MobileNet 等。有的网络结构从加深网络层数的角度出发进行设计 (ResNet), 有的则考虑加宽网络结构增强并行能力 (Inception)，还有的为了应用而进行更有效的卷积操作 (MobileNet) 提高速度。这些网络都将图像识别的性能提高到了超越人类的水准。经过第二周项目的练习，我们已经掌握了如何自己设计并实现一个网络，本次实验我们将使用 Pytorch 已经搭建好的上述各种网络，进一步增强我们之前的分类网络的性能。

2. 数据增强技术是一种正则化的方式，它也是一种工程技巧，可以让大多数网络的泛化性能进行提升。本次实验我们同时会使用一个数据增强库，帮助我们增强数据模式，提升模型的泛化性能。

3. 你将练习实现一个基于常用 CNN 的图像分类器，并且采用数据增强技术，将你在第二周项目上得到的准确精度大幅度提高。

2 开发环境配置

1. 激活之前的 vclab 环境

conda activate vclab

2. 安装 imgaug (一键安装下载最新版本就好)

pip install imgaug

*本工程由 IAIR Visual Computing Laboratory 参考斯坦福 CS231n 设计，有问题请联系 lrj466097290@stu.xjtu.edu.cn。

3 任务内容

本工程有 5 个部分。你将使用高层的 Pytorch API，调用现成的刚才提到的各种网络，感受成熟网络的威力。然后将使用 Torchvision 模块的 transforms 子模块，应用各种图像增强方式，在固定一个网络结构的基础上，将增强前后的结果进行对比。

1. 第一部分，准备：我们将继续使用 CIFAR-10 数据集。

2. 第二部分，调用现成的常用网络。常用网络的设计背后通常包含着一些结构设计的思考，请参阅 `http://cs231n.stanford.edu/slides/2020/lecture_8.pdf` ，对这些思考进行了解，以帮助未来对网络结构进行创新。

3. 第三部分，调用现成的图像增强方法。图像增强是非常常用且重要的工程技巧，通常使用了增强技术的效果会比不使用高许多，甚至收敛但无法泛化的网络一旦经过数据增强技术，泛化性能会非常优秀。这些提升泛化性能的正则化技术还有许多，请参阅 `http://cs231n.stanford.edu/slides/2020/lecture_9.pdf` ，了解更多的泛化技术，为未来网络的训练提供帮助。

4. 第四部分，自学 imgaug，对图像进行更加灵活多变的增强。Torchvision 的增强模块比较少，而且增强过程的代码难以学习和造轮子。推荐 imgaug 这个库，直接对 numpy 类型进行操作，非常适合学习常用的图像裁剪，对称，仿射等操作，请参阅 `https://github.com/aleju/imgaug` 进行学习。

5. 第五部分，CIFAR-10 开放式挑战：请在 CIFAR-10 上通过调用成熟网络和图像增强技术，以获得尽可能高的精度。你可以用任何现有的成熟网络体系，优化器，调整超参数或其他高级功能进行实验。

4 实验

1. 第一部分：直接运行 python part1.py, 下载 cifar10 数据集，下载完成后，包含数据的文件夹'cifar-10-batches-py' 将出现在 vclab/datasets/ 路径下；同时输出的'using device:' 将告诉现在使用的是 CPU 还是 GPU 版本的 pytorch（对最终精度无影响，但请大家尽力安装 GPU 版本，飞速训练提升调参效率）。

2. 第二部分：

(1) 在 part2.py line 60 使用 resnet18 网络, 直接运行 python part2.py, 无须修改任何超参数，**测试集**正确率在 75% 以上。

(2)正确理解 `https://pytorch.org/docs/stable/torchvision/models.html` 的写法后，尝试使用其他的网络模型，运行 python part2.py，无须修

改任何超参数, 记录你达到的**测试集**正确率 (请至少尝试 10 种网络模型)。

3. 第三部分:

(1) 在 part2.py line 27-31, 取消注释, 直接运行 python part2.py, 在添加了图像补零, 图像随机水平翻转, 图像裁剪, 图像 tensor 化 (Pytorch 的操作对象是 tensor 不是 numpy 类型，这两者在此方法处进行了类型转换) 无须修改任何超参数，**测试集**正确率在 85% 以上。

(2) 正确理解 `https://pytorch.org/docs/stable/torchvision/transforms.html` 的写法后，尝试使用其他的图像增强方法，运行 python part2.py，无须修改任何超参数, 记录你达到的**测试集**正确率 (请至少尝试 10 种增强方法)。

4. 第四部分

(1) 正确理解 `https://github.com/aleju/imgaug` 的教程，请将 cifar10 的一张图片读入为 numpy 类型，然后使用 imgaug 的图像增强方式对该图片进行操作后，可视化增强前后的图片。

(2) 请将 cifar10 的多张图片读入为 numpy 类型 (比如 batchsize=10), 对 10 张图片使用 imgaug 进行图像增强，可视化增强前后的图片组。

5. 第五部分

在 part2.py 尽情发挥，使用任何模型，任何图像增强方式, 也可以对优化器、超参数等进行修改，请尽量达到高的精度, 看看自己可以在全世界的排名榜单中取得如何的成就。`https://benchmarks.ai/cifar-10`, `https://paperswithcode.com/sota/image-classification-on-cifar-10`。

5 报告要求

第二、三部分请以**表格**的形式告诉我

- 你使用的模型 (图像增强方法)
- 你达到的精度

第四部分请以**图片**的形式告诉我

- 你使用的 imgaug 图像处理方法
- 处理前的图像
- 处理后的图像

第五部分请介绍你最终的方法，使用了哪一种网络，使用了那些图像增强方式的排列组合，获得了多少的精度。**请在提交报告的时候附上你的代码，我会运行一遍来确认你的结果是否和报告一致**。