

实验三：Kaggle 分类比赛

X

日期：July 15, 2022

摘 要

本实验为模式识别与深度学习课程的实验三：Kaggle 分类比赛，主要任务是利用 Kaggle 数据集完成 Plant Seedlings Classification。在本实验中，我们小组基于 PyTorch 和 Cuda 实现了 VGG/ResNet/SENet 三种网络结构，并在 ResNet 网络结构的基础上进行了性能调优。我们的基础 ResNet 网络最终在 Kaggle 所提供的数据集上得到了 0.93 的 score，在自选性能调优部分，为我们将网络更换为 CoAtNet 之后，我们进一步取得了 0.95 的 score。

关键词：Kaggle, Plant Seedlings Classification, VGG, ResNet, SENet, CoAtNet

1 深度学习框架与实验环境

本实验采用的深度学习框架是 Pytorch。另外，由于 Kaggle 提供了每周 30h 的在线免费算力，因此整个实验在 Kaggle 的 Online Notebook 中完成。

2 实验背景知识

2.1 VGG 简介

VGG 在 2014 年由 Oxford Visual Geometry Group 和 DeepMind Co. 联合提出。其意义在于其证明了增加网络的深度能够在一定程度上影响网络最终的性能。VGG 有两种结构，分别是 VGG16 和 VGG19，两者的区别只是网络深度上存在差异。

从某种程度上来说，VGG 可以看成是加深版的 AlexNet，采用连续的几个 3×3 的卷积核代替 AlexNet 中的较大卷积核。通过反复堆叠 3×3 的小卷积核和 2×2 的最大池化层，VGGNet 成功的搭建了 16-19 层的深度卷积神经网络。与之前的网络结构相比，性能得到了较大幅度的提升；榆次同时，VGG 的泛化能力非常好，在不同的图片数据集上都有良好的表现。到目前为止，VGG 依然经常被用来提取特征图像。

VGG 的网络结构如下：

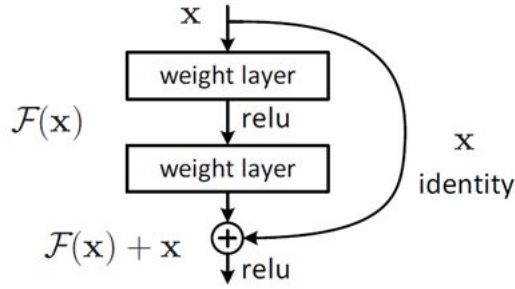
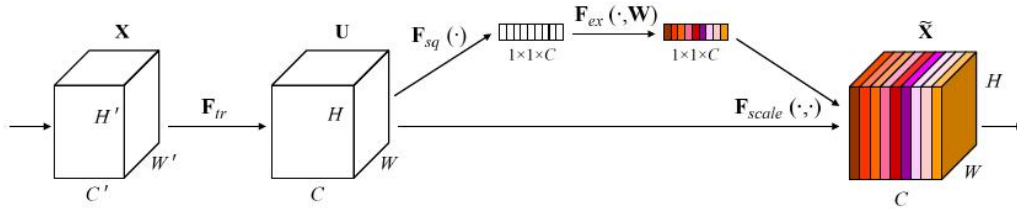


图 3: ResNet 中的基本单元

2.3 SE-Net 简介

对于 CNN 网络来说，其核心计算是卷积算子，通过卷积核从输入特征图学到新特征图。从本质上讲，卷积是对一个局部区域进行特征融合，这包括空间上（W 和 H 维度）以及通道间（C 维度）的特征融合，而 SENet 网络的创新点在于关注 channel 之间的关系，希望模型可以自动学到不同 channel 特征的重要程度。为此，SE-Net 提出了 Squeeze-and-Excitation Block (SE Block)，如下图所示。SE-Net 的中心思想是，对于每个输出 channel，预测一个常数权重，对每个 channel 进行加权，本质上，SE 模块是在 channel 维度上做 attention 或者 gating 操作，这种注意力机制让模型可以更加关注信息量最大的 channel 特征，而抑制那些不重要的 channel 特征。SE-Net 的另一个优点是可以很方便地集成到现有网络中，从而提升网络性能，并且代价很小。



A Squeeze-and-Excitation block.

图 4: SE Block 的结构

3 实验过程

3.1 概述

为了对比不同模型的效果，在本实验中，不同模型的超参数均保持一致（如 batch size 均设置为 16）。另外，在 3.4, 3.5 两节中，我们小组基于 ResNet 结构进行修改，以对比不同方法的最终效果。

3.2 使用 CUDA 与 argparse

在本实验中，按照实验要求使用 CUDA 与 argparse，CUDA 的使用通过设定 device 为 GPU 完成，argparse 的相关代码如下：

```

parser.add_argument('--gpu', type=str, default='0', help='gpu')
parser.add_argument('--data_path', type=str, default='.././datasets/caltech-101/101_ObjectCategories',
                    help='path to train set')

parser.add_argument('--save_dir', type=str, default='working/checkpoint', help='save dir')
parser.add_argument('--log_dir', type=str, default='working/log', help='log dir')
parser.add_argument('--save_prefix', type=str, default='working/ResNet18_SGD_e', help='save prefix')

parser.add_argument('--lr_initial', type=float, default=1e-4, help='initial learning rate')
parser.add_argument('--weight_decay', type=float, default=2e-5, help='weight decay')

parser.add_argument('--batch_size', type=int, default=16, help='batch size')
parser.add_argument('--epoch_num', type=int, default=60, help='epoch num')

parser.add_argument('--checkpoint_frequency', type=int, default=3, help='checkpoint frequency')
parser.add_argument('--test_num', type=int, default=2, help='test num')
opt = parser.parse_args(args=[])

```

图 5: argparse 部分代码

3.3 测试不同模型的效果

在本实验中，我们基于 PyTorch 实现 VGG-11，ResNet-18，以及 ResNet-18 + SE block 三种网络的结构，并在 Kaggle 上进行了提交。这三种模型的表现如下：

Submission and Description	Private Score	Public Score	Use for Final Score
ResNet_SE.csv just now by Minco Lee ResNet18 + SE Block	0.92947	0.92947	<input type="checkbox"/>
ResNet.csv a few seconds ago by Minco Lee ResNet18	0.93198	0.93198	<input type="checkbox"/>
VGG.csv a minute ago by Minco Lee VGG11	0.92947	0.92947	<input type="checkbox"/>


图 6: 三种模型的得分

可以看到，提交至 Kaggle 进行评估，得到的结果是：原始的 ResNet 效果最好，而 ResNet + SE Block 与 VGG 的效果次之。

3.4 测试不同优化器的影响

在原本的 ResNet 实现中，我们所使用的 Adam 优化器。在参数不变的情况下，我们将其替换为 SGD 优化器，实验结果如下：

YOUR RECENT SUBMISSION



ResNet18_SGD.csv
Submitted by Minco Lee - Submitted just now

Score: 0.80226

↓ Jump to your leaderboard position

图 7: 使用 SGD 的 ResNet 结果

可以看出，在换用 SGD 优化器后，性能有了明显的下降。但在训练过程中可以观察到，直到 30 个 epoch 训练完毕，网络的性能仍有提升。说明也可能是训练不够充分的原因，为此，我们增大训练 epoch 数至 60，再次进行实验，得到了 5 个点的性能提升：

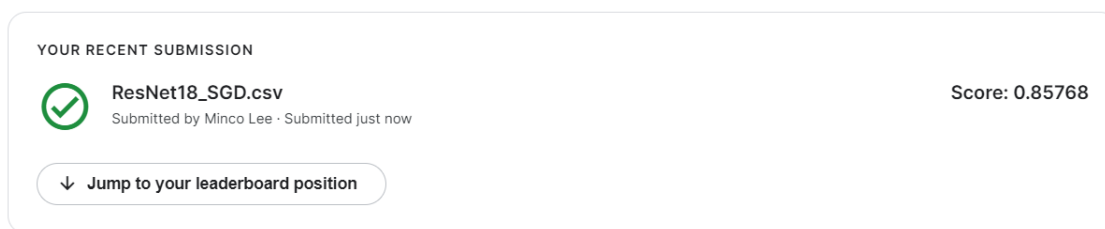


图 8: 使用 SGD 的 ResNet 结果 (训练 60 个 epochs)

3.5 测试不同增广方法的影响

3.5.1 水平翻转与垂直翻转

在本实验中，我们小组测试了不同 data augmentation 方法对结果的影响。具体而言，在原本的 ResNet 中，我们对图片按 0.5 的概率进行水平翻转，将其改为按 0.5 概率进行垂直翻转后，结果如下：

Submission and Description	Private Score	Public Score	Use for Final Score
ResNet18_vertical.csv 16 hours ago by Minco Lee ResNet18 Vertical	0.91561	0.91561	<input type="checkbox"/>

图 9: ResNet 垂直翻转结果

可以看出，在将水平翻转改为垂直翻转后，score 有了小幅度的下降。

3.5.2 旋转

将 data transform 中将翻转操作改为 rotation 旋转操作，得分有微小幅度的提升，如图所示：

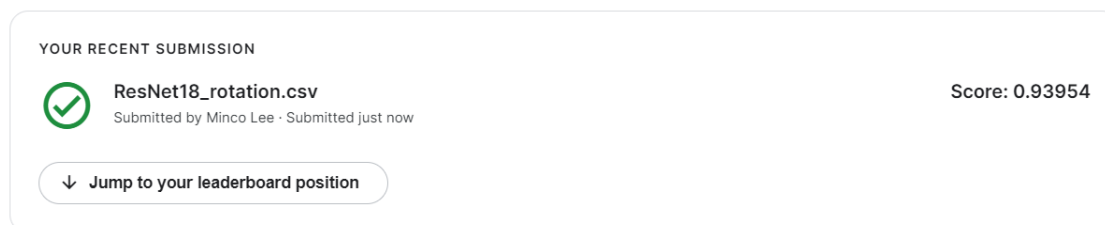


图 10: ResNet 旋转结果

3.6 自选改进方式

为了进一步提高在该 Kaggle 竞赛上的得分，我们选择将模型更换为 CoAtNet，并取得了较好的效果。

3.6.1 模型简介

尽管 Transformers 具有更强的 model capacity (模型能力), 但因为缺乏 inductive bias (归纳偏置) 特性, 它的泛化性要落后于 CNN。而 CoAtNets 有效地结合二者的长处。它的构建主要基于两个关键想法: (1) 我们可以通过简单的 relative attention (相对注意力) 将 depthwise Convolution (深度卷积) 和 self-Attention (自注意力) 自然统一; (2) 在提升泛化性、能力和效率方面, 按照一定的原则, 垂直摆放卷积层和注意力层会非常有效。在原论文中, CoAtNets 在多个数据集上取得了 SOTA 的效果。

3.6.2 实验效果

如下图所示, 可以看到, 在换用 CoAtNet 之后, 我们得到了 0.955 左右的评分, 为本实验中我们小组的最高得分。

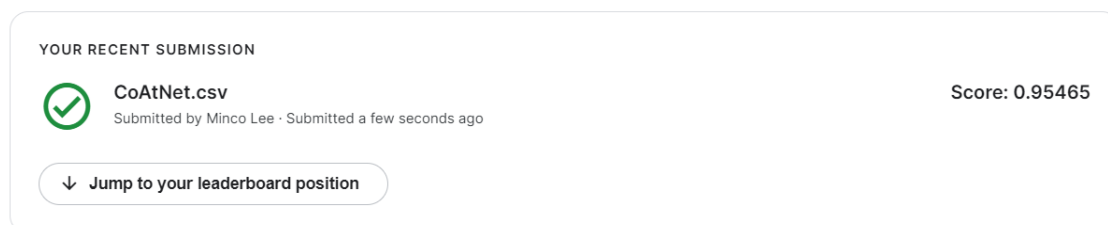


图 11: CoAtNet 评估结果

4 成员分工

- 姚舜宇: 构建 ResNet 与 ResNet + SE Block 模型, 完成性能调优公共部分, 自选部分与模型训练过程。
- 江骏朋: 完成数据集构建, 实现 VGG 网络模型, 完成性能调优公共部分与自选部分 (CoAtNet), 负责模型评估与调优。
- 李旻翀: 完成性能调优公共部分, 负责模型评估调优与报告撰写。

5 说明文档

在实验文件夹中, ./实验结果下保存着每个模型的.pth 模型文件与.csv 评估文件。根目录下的 dl-lab3.ipynb 为本次实验的代码文件。