中山大學

Computer Vision

By Mindspore

题　　目：	Plant Pathology-2021分类任务
授课教师：	梁小丹
姓　　名：	孙广岩
学　　号：	20354242
日　　期：	2022/10/15

Plant Pathology-2021(By Mindspore)

孙广岩，20354242

中山大学，智能工程学院

摘　要： 本次作业是一个图片的多标签分类任务，我使用了one hot encoding来对标签进行了编码，之后将数据集转换为了更高速的MindRecord格式。在之后的实验中，我尝试了加入了新的Loss函数，并且。经过实验，我在测试集达到了的准确度。最后我对模型进行了一些可视化分析，来使模型更具有可解释性。 本次的实验软件环境为Mindspore v1.8.1，硬件环境为RTX 3090 * 1。

关键词： 深度学习；Mindspore；细粒度分类；可解释性分析

1 Dataset

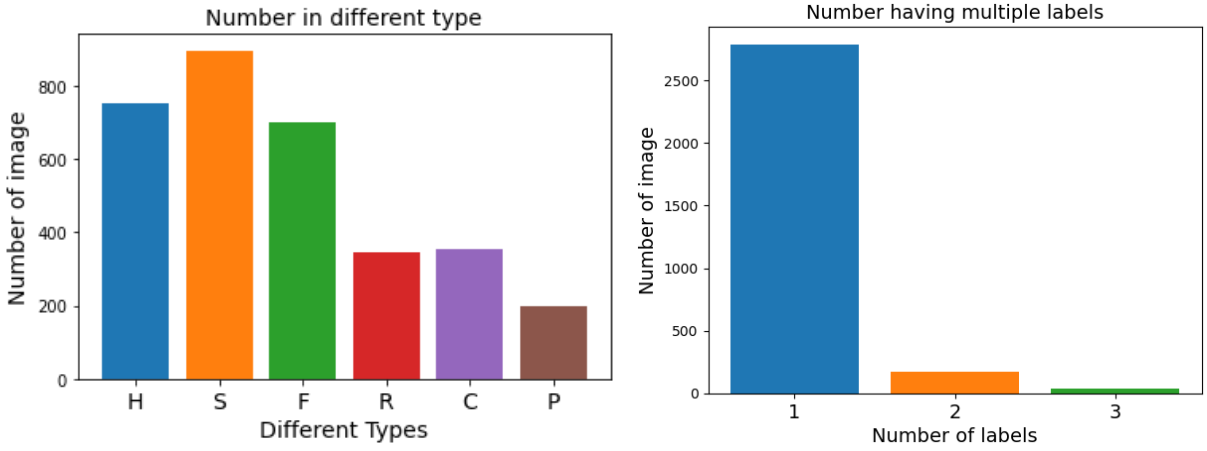
Plant Pathology-2021是Kaggle举办的FGVC8(CVPR2021-workshop)的数据集，任务是对叶片的病害进行分类，它有两个难点，一个是叶片病害分类的类内差异大，类间差异小，属于细粒度分类分类问题，之后也会针对此进行一些改进，另一个难点为不同的类别之间有重复，这会影响模型的训练精度，这里将类别改为6类，转换为了多标签分类问题。

2 Building Pipeline

这个部分我按照Kaggle上的“传统”，是由notebook来实现的（见代码文件中的*simple_pipeline.ipynb*），之后的模型改进时的训练会构建一个相对完整的代码项目，下面也是简单的介绍一下，notebook本身我认为还是写的比较详细的，这只是进行了一些数据上的分析，并构建了一个可以简单的迁移学习Mindspore模型。

2.1 Data Analysis

首先我对训练数据集进行了简单的一个分析，我首先统计了各个不同种类叶子的数量（左图），可以看到整个数据集的标签分布并不是非常的平均，前面的三种会多一些。之后我又统计了数据集中多个标签的数量（右图），可以看到一个标签的数量占有绝对的优势，同时有三个标签的其实是非常少的。

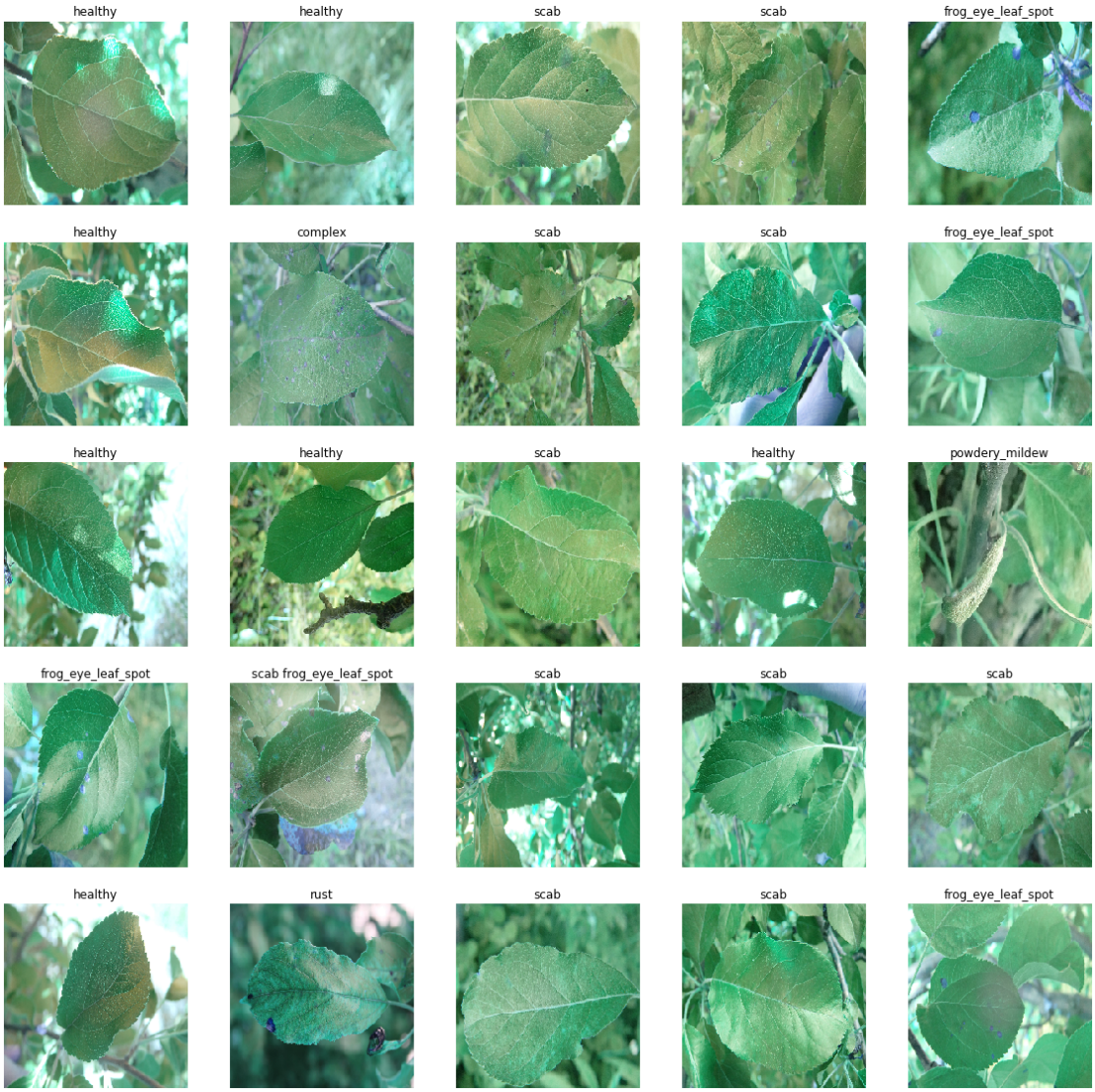


2.2 Data Loader

Data Loader 这个部分我首先参考了几篇文章：一个是 Kaggle 上关于这个数据集在 Tensorflow 上的读取方法[1]，一个是 Mindspore 官方的构建自定义数据集的方法[2]，但可惜官方构建的自定义数据集经过测试比较缓慢，成为了训练速度的瓶颈，原因主要是任务提供的数据集图片的分辨率很高，导致读取图片很慢。

所以我把数据集中所有的图片变为256x256的大小并且构建好one_hot的标签，变为Mindspore 官方支持的MindRecord 格式[3]进行读取，转换格式的代码可以在 *src/create_mindrecord.py* 查看。最终全部的数据总占大约177MB，可见*plant_dataset_mdb*文件夹。

简单在Notebook展示一下数据集的图片与标签：



2.3 Simple CNN(Resnet)

因为这里我只是想先构建一个pipeline，所以我先选择了一个简单的模型，这样会比较省时间。这里我使用了Mindspore 官方提供的 Vision 套件 [4] 使用 Resnet50 来进行一个迁移学习。

2.4 Simple Train and Inference

由于ResNet50中的预训练模型是针对ImageNet数据集中的1000个类别进行分类的，在本次作业中只有6种，所以需要重置预训练模型中的分类器，然后重新微调网络。

然后选择优化器和损失函数，这里优化器就直接选择了Momentum，损失函数这里参考了使用了多标签分类中最常用的BCE Loss，对于评估指标来说选择的就是比较基础的准确度，也就是只有当预测与实际完全一致的时候，才可以当作正确的。

之后 Mindspore 训练的方法非常简单，可以直接调用 Mindspore 的 Model 这个 API 来进行训练，使用MindSpore Vision中的 `mindvision.engine.callback.ValAccMonitor` 接口（仅支持GPU）打印训练的损失值和评估精度，且在训练完成后，保存评估精度最高的ckpt文件(best.ckpt)到当前目录下。那么我也简单的进行了一下训练。把刚才训练的模型来进行一下测试，可以看到如下结果：

Model	Val Acc	Test Acc
Resnet50	0.858	0.814

3 Performance Enhancement

介绍完了一个简单的训练流程，现在开始做一些调优的工作。这里因为VIT的实验开销比较大（速度会慢不少），所以还是在Resnet50上面进行的一些调优，VIT模型只是跑了一个基础的BCE loss的版本。

3.1 VIT Model

首先尝试了一下vision transformer的效果，这里选择了使用vit_base_16模型，结果在后面的表格有展示。可以看到最终的测试集准确率提升了约3个点，说明了vision transformer的效果相对于Resnet50来说还是要好不少的（当然训练时间也变长了一些）。

3.2 Inspiration from fine-grained classification model

我尝试了Focal Loss，这是目标检测领域常用的一个损失函数，由Kaiming He在RetinaNet[5]这篇论文中提出，主要是针对样本不均衡问题的，Focal Loss 的具体形式为：

$$L_{fl} = -(1-\hat{p})^{\gamma} log(\hat{p})$$

主要的原理还是减少置信度很高的样本损失在总权重的比重。最终效果可以看到在Resnet50上测试集还是有一定的提升的。

最终的一些结果汇总：

Model	Val Acc	Test Acc
Resnet50	0.858	0.814
Resnet50 wF	0.859	0.835
Vit	0.859	0.840

可以看到经过引入，模型有了一定的提升，当然可能因为整个数据集比较小的原因，所以整体模型提升就不是很明显了。

4 Visualization Analysis

这里进行一些简单的可视化分析。MindSpore XAI[6]是一个基于昇思MindSpore的可解释AI工具箱。对于CNN的分析可以使用这个工具包中的GradCam[7]方法，它是一个经典的基于梯度的解释器。对于VIT来说最经典的方法是Attention Rollout[8], 由于有些算子在Mindspore没有实现（具体来说，在实现Attention Rollout时，我们需要使用topk函数来取最小的几个数据，在Mindspore中对应的函数只能取最大不能取最小，我也没有特别找到很好的替代函数），所以这里也就是附上一个基于Torch的一个简单的示例，使用的代码来自Github。

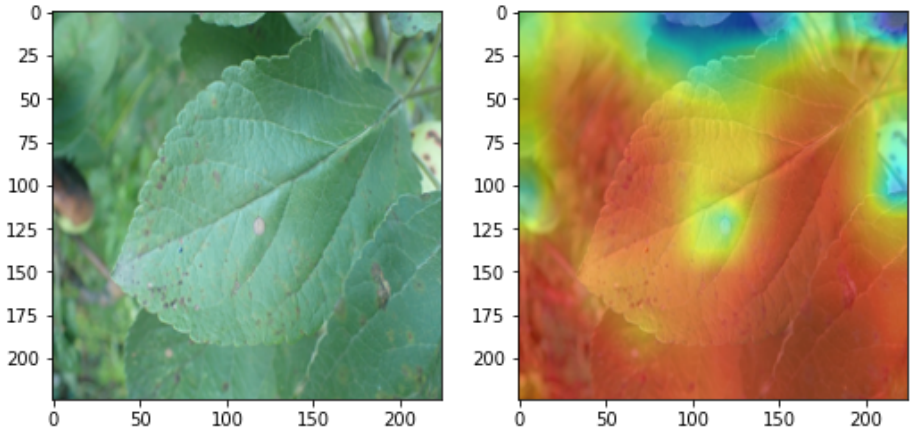
4.1 Resnet

使用Mindspore XAI 分析我训练好的一个Resnet50模型，可以看到如下图：

左：原图；中：标签为scab的结果；右：标签为frog_eye_leaf_spot的结果

这张图片是一个多标签的图片，可以看到对于不同的标签，Resnet关注了不同的染病区域。

4.2 VIT



左：原图；右：Attention Rollout 结果

可以看到Vit还是关注到了叶片靠近边缘的染病区域，还是有着不错的效果的。

5 Conclusion

综上所述，本次任务还是非常有趣的，我学习到了如何使用 Mindspore 来构建一个完整的图像分类任务项目，也更加深入的了解了细粒度分类这个研究课题。这中间也阅读了一些这方面的工作，虽然有一些工作可能Mindspore不是很好实现，所以没有办法用到作业中，但是最终尝试了一个实现比较简单的方法，最终还是有一定的效果，对于我之后的学习与研究也提供了许多启发。

我在构建数据集这里一开始消耗了比较多的时间，根据网站的教程写出可以运行的代码并不困难，但是在我使用的时候发现数据集的读取速度过于缓慢，导致成为我训练作业时的一个时间瓶颈，这个主要还是数据集中图片分辨率比较高导致的，所以我将数据集转换为Mindspore支持的MindRecord格式，这样读取的速度就变得正常了。之后就是正常的一个训练流程，官网也有相应的教程，我是用了GPU平台，因为Mindspore的Vision套件是目前只支持GPU平台的，这样的话代码会比较简洁很多，同时实验也会更加高效。

之后的调优工作我也是看了一些关于细粒度分类的综述来寻找有没有合适的方法，这里我主要看的是弱监督，因为我们只有一个类别标签，没有bounding box之类的，所以强监督的参考价值可能就没有这么高了。经过阅读了一些细粒度分类相关的论文，我读到了一篇来自ECCV2018的一篇文章[9]，作者提出了一种可端到端进行训练的优化方法----Pairwise Confusion, 在激活中故意引入混淆来减少过拟合。通过实验证明PC能够提高定位能力并且在六个细粒度任务数据集上实现了最佳的性能。在训练阶段不需要额外的参数调整，在测试阶段不会增加额外的耗时，PC是很容易应用的。模型结构大概是这样

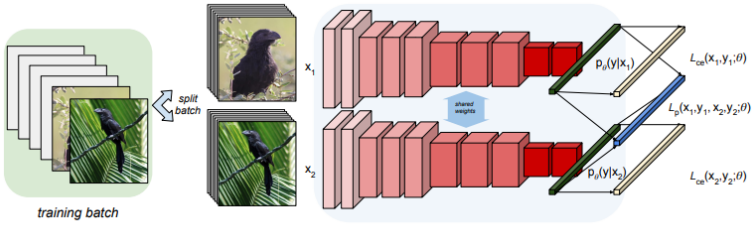


Fig. 1. CNN training pipeline for Pairwise Confusion (**PC**). We employ a Siamese-like architecture, with individual cross entropy calculations for each branch, followed by a joint energy-distance minimization loss. We split each incoming batch of samples into two mini-batches, and feed the network pairwise samples.

但是我尝试过后发现最终的准确率反而变低了，我同样在他们的Github网址中也发现了有人也提出了issue说不能使模型准确度得到提升，这个还是比较奇怪的，可能还是有些setting他们没有公布，但是他们的实验结果确实还是不错的。

之后我尝试了对模型进行了一定程度上的可视化分析。我是首先在网上搜集了资料，了解到了GradCAM和Attention Rollout的方法，然后发现Mindspore官方提供了一个可视化包可以进行GradCAM, 但可惜Attention Rollout的源码我再转换过程中发现算子不是很适配，就没有再继续改了。不过我也看了一下他们的论文，了解了整个他们可视化的原理，还是非常有趣的。

参考文献与资料:

[1] Experiment Tracking with Weights and Biases | Kaggle

[2] MindSpore 自定义数据集

[3] MindRecord数据集格式介绍

[4] Mindspore Vision

[5] Focal Loss

[6] Mindspore XAI

[7] Grad-CAM

[8] Attention Rollout

[9] Pairwise Confusion for Fine-Grained Visual Classification