

Plant Pathology 2021 with MindSpore

fanggan@mail2.sysu.edu.cn, 方桂安

摘要—本次作业目标是学会使用 mindspore 进行简单卷积神经网络的开发，对 Plant Pathology-2021 数据集进行分类。我通过五折交叉验证、软化标签、数据增强、模型融合等方法，最终在测试集上的准确率达到了 90.5%。

关键词—MindSpore, 细粒度分类, 多标签, FGVC8

I. 概述

PLANT PATHOLOGY 2021 数据集是计算机视觉和模式识别会议 CVPR 2021 上细粒度视觉分类 FGVC8 研讨会的一部分，本次作业中的训练集共 3000 张，测试集和验证集均为 600 张，对应的 csv 文件中记录了图片的 id 和标签，共 12 类。但实际上该数据集中只有 6 类标签，分别为 healthy, scab, frog_eye_leaf_spot, rust, complex 和 powdery_mildew，其余几类为多标签图片。

II. 分析

该任务是要识别苹果树叶是否有疾病以及相应的疾病种类（同一个叶子可能有多种疾病），该任务的难点主要包括：

可能存在脏数据，人工也难以判别导致 gt 错误；多标签数据的严重缺少，最少的类别与最多的类别数据量，比例达到 1:50 以上；叶片病害分类的类内差异大，类间差异小……

开始上手该任务前，我对数据进行了简单的分析：

数据集中有五个疾病类别以及 healthy 类别，这意味着存在 2 到 5 个可能的组合。尽管并非所有组合都是存在的，但这个训练集只包含其中的 11 个。测试集中可能有其他组合不在训练集中。因此，我打算以多标签的方式处理该问题。

我将 train 和 val 中的图片进行了合并，共计 3600 张图片，其中的标签分布如下，可以看出总数并不是 3600，验证了上述的猜想，存在脏数据，但是由于数据量较少，我决定不去除脏数据，而是通过软化标签的方式来处理。

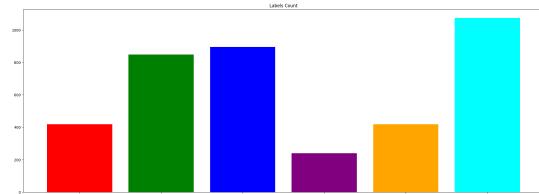


Fig. 1: 数据分布

表 1 标签数量分布

id	label	count
1	complex	418
2	frog_eye_leaf_spot	849
3	healthy	895
4	powdery_mildew	240
5	rust	418
6	scab	1073

III. 数据样本

A. Healthy

没有任何疾病的叶子。



Fig. 2: healthy

B. Scab

scab 是一种细菌或真菌植物病害，其特征是果实、叶子和茎上的硬皮病变。

C. Frog_eye_leaf_spot

frog_eye_leaf_spot 是由真菌 Cercospora sojina 引起的。这种疾病发生在美国各地和加拿大安大略省。



Fig. 3: scab



Fig. 7: complex



Fig. 4: frog_eye_leaf_spot

当在田间广泛分布时，蛙眼叶斑病会导致严重的产量损失。叶子上的病斑很小，形状不规则到圆形，灰色，有红褐色的边缘。

D. Rust

rust 是一种由真菌引起的植物病害。锈病通常以黄色、橙色、红色、锈色、棕色或黑色粉状脓疱形式出现在叶子和果实上，影响许多经济上重要的植物物种。

E. Powdery_mildew

丹毒目中的许多真菌种类会引起白粉病。白粉病的症状非常明显，使其成为最容易识别的疾病之一。受感染的植物的叶子和茎上有白色粉状斑点。



Fig. 5: rust



Fig. 6: powdery_mildew

F. Complex

如竞赛所述：具有太多疾病而无法直观分类的不健康叶子将具有该 complex 类别，并且还可能具有已识别的疾病子集。

IV. MINDSPORE

正式进行训练之前，我还先进行了一段时间的研究，了解了 mindspore 的一些基本 API，以及如何使用 mindspore 进行训练。

在这部分里，我将该任务暂时定为一个 12 分类任务，使用 lenet5, vgg16, resnet50, vit 分布训练了 40 个 epochs，损失函数为 SoftmaxCrossEntropyWithLogits（使用 one-hot 编码获取预测值和真实之间的 softmax 交叉熵。），优化器为 Momentum。由于目的是掌握该框架的使用，故没有过于追求模型的性能。

图片展示的是 40 个 epoch 训练时模型在训练集上的准确率，其中 lenet 没能拟合，图片尺寸为 32x32 损失了大量的信息；其他三个模型训练的图片尺寸为 224x224，都能拟合到一个较好的结果，特别是 resnet50，很快就学习到了训练集中的特征。

表格中展示的是各模型在当前状态下的测试集准确率，可以看出在没有使用任何技巧，简单地做一个 12 分类的任务是不可行的。

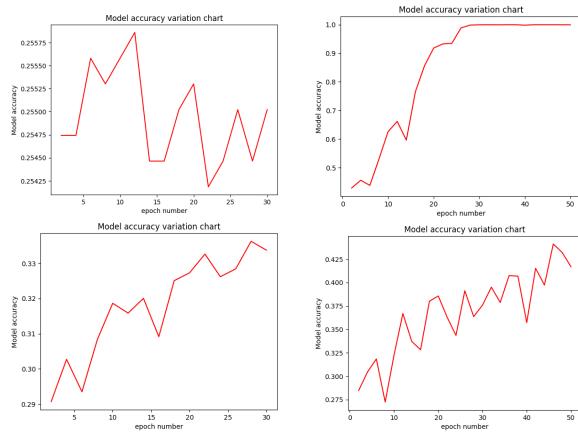
表 2 测试集准确率

num	model	acc
1	lenet5	1.04%
2	vgg16	8.85%
3	resnet50	33.50%
4	vit	26.38&

V. 模型选择

除了 ResNet50, Vit 外，我先使用了一些在比赛中广泛使用的预训练 CNN，分别是：

- EfficientNet-B4
- ResNext50
- SEResNeXt50



(a) 上:lenet5, 下:vgg16 (b) 上:resnet50, 下:vit_1_32

Fig. 8: 训练过程准确率变化

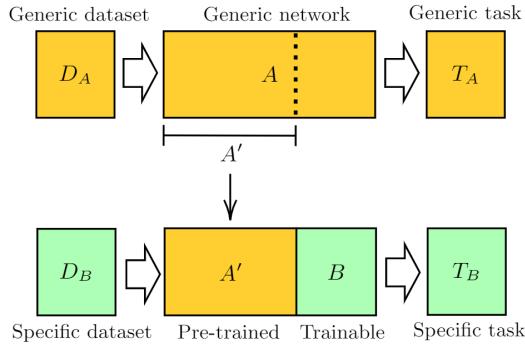


Fig. 9: 迁移学习

A. 迁移学习

在机器学习中，迁移学习是指存储在解决一个问题时获得的知识，并将其应用于另一个相关问题。预训练模型是针对与当前任务不同的任务进行训练的，但它提供了一个非常有用的起点，因为在训练旧任务时学习的特征在训练新任务时很有帮助。

与 mindspore 相关的生态中 MindSpore ModelZoo 与 MindSpore Hub 在本次作业中给我提供了很大的帮助，其中 MindSpore ModelZoo 提供了模型的完整代码，而 MindSpore Hub 则提供了模型的预训练权重。

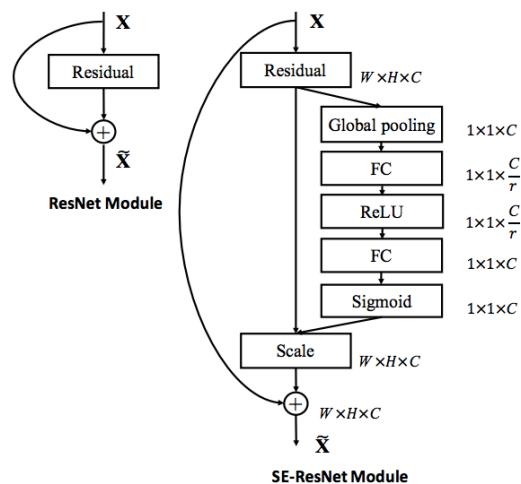
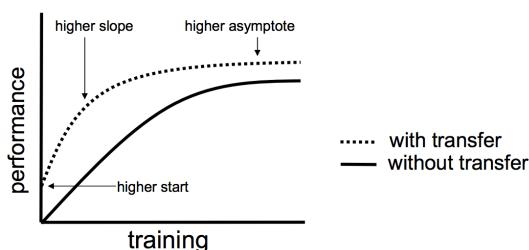


Fig. 10: Module 对比

- 更好的初始模型：迁移学习是在没有任何知识的情况下构建模型的更好起点，即使没有训练也可以执行一定级别的任务。
- 训练后更高的准确性：迁移学习允许机器学习模型在更高的性能水平上收敛，从而产生更准确的输出。
- 更快的训练：通过使用预训练模型，训练可以比传统方法更快地达到预期的性能。
- 对抗过度拟合：迁移学习的主要优势在于它降低了训练数据不足相关的问题影响，这是过拟合的原因之一。

VI. SEResNeXt50_32x4D

经过对比实验，我最终选择了 SEResNeXt50_32x4d 作为最终的模型。

SEResNext 是带有 Squeeze-and-Excitation 模块的 ResNext-50。Squeeze-and-Excitation Networks (SENNets) 为 CNN 引入了一个构建块，它几乎没有计算成本就可以改善通道的相互依赖性。除了这种巨大的性能提升之外，它们还可以轻松添加到现有架构中。

其中前缀 se 为 squeeze and excitation 的过程，这个过程的原理为通过控制 scale 的大小，把重要的特征增强，不重要的特征减弱，和 attention 原理相同，目的是让提取的特征指向性更强，从而能更好的对 FGVC(Fine-Grained Visual Categorization 细粒度视觉分类) 任务中精细的特征做识别。

VII. 训练配置

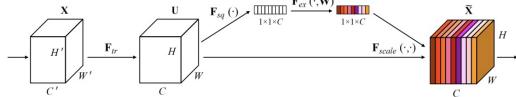


Fig. 11: SEResNeXt50_32x4d

A. Adam 优化器

Adam 优化器是深度学习中最常用的优化器之一。使用 Adam 训练时，模型的收敛速度通常比使用常规的随机梯度下降法(SGD)要快得多，而且与带 momentum 的 SGD 相比，Adam 通常需要更少的学习率调整。Adam 对动量 SGD 的改进在于（除了 momentum）还计算每个被调整的参数的自适应学习率。这意味着在使用 Adam 时，与使用 SGD 时相比，在训练过程中修改学习率的需要更少。

经过实验测试，我使用的优化器是 Adam，相较于其他优化器，Adam 配合其他设置在该任务上的表现最优。

Adam 的更新公式为：

$$\begin{aligned} m_t &\leftarrow \beta_1 m_{t-1} + (1 - \beta_1) \cdot g_t \\ v_t &\leftarrow \beta_2 v_{t-1} + (1 - \beta_2) \cdot g_t^2 \\ \hat{m}_t &\leftarrow \frac{m_t}{1 - \beta_1^t} \\ \hat{v}_t &\leftarrow \frac{v_t}{1 - \beta_2^t} \\ \theta_t &\leftarrow \theta_{t-1} - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \end{aligned}$$

其中 α, β_1, β_2 和 ϵ 是标量超参数。 m_t 和 v_t 是一阶和二阶矩。 \hat{m}_t 和 \hat{v}_t 是有偏修正的矩。 ϵ 用于修正除以零的误差，但也作为一种超参数的形式，对梯度的差异起作用。假设 $\epsilon = 0$ 所采取的有效步骤是，

$$\Delta t = \alpha \cdot \frac{\hat{m}_t}{\hat{v}_t}$$

这是有界的，

$$|\Delta t| \leq \alpha \cdot \frac{1 - \beta_1}{\sqrt{1 - \beta_2}}$$

当 $1 - \beta_1 > \sqrt{1 - \beta_2}$ 和

$$|\Delta t| \leq \alpha$$

否则，在最常见的情况下，

$$|\Delta t| \approx \alpha$$

B. MultiClassDiceLoss

原本使用的 SoftmaxCrossEntropyWithLogits 虽然使用了 one-hot 编码，但是在训练过程中，模型的 loss 值一直在波动，而且在验证集上的表现也不理想，且我打算采取的是多分类的方式，即 Multi-hot，故最终选择了 MultiClassDiceLoss 作为损失函数。

对于多标签问题，可以将标签通过 one-hot 编码转换为多个二分类标签。每个通道可以看做是一个二分类问题，所以损失可以通过先计算每个类别的二分类的 DiceLoss 损失，再计算各二分类损失的平均值得到。

Dice 系数是一个集合相似性 loss，用于计算两个样本之间的相似性。当分割结果最好时，Dice 系数的值为 1，当分割结果最差时，Dice 系数的值为 0。

Dice 系数表示两个对象之间的面积与总面积的比率。函数如下：

$$\text{dice} = 1 - \frac{2 * |\text{pred} \cap \text{true}|}{|\text{pred}| + |\text{true}| + \text{smooth}}$$

pred 表示 logits，true 表示 labels。

其实在测试的过程中我更倾向于使用 Focal Loss (解决类不平衡问题) 或者 ArcFace Loss (解决细粒度分类问题)，但是两者都没能在我的模型上收敛，且我对于 mindspore 框架的掌握还不够，所以最终只能放弃。

C. 训练参数

其他方面的参数设置如下：

- batch_size: 128
- learning_rate: 0.001
- image_size: 224 x 224
- epochs: 50
- threshold: 0.55
- 1x1 Convolution, Batch Normalization, Convolution, Grouped Convolution, Global Average Pooling, ResNeXt Block, Residual Connection, ReLU, Max Pooling, Softmax

VIII. 数据增强

由于数据集相对较小，直接使用原始数据进行训练会导致模型存在过拟合的风险，为了更好的增加模型鲁棒性，我对数据集进行了一系列增强操作来扩充原始数据集。

除了常规的 Resize, Normalize 还使用了以下的技巧：

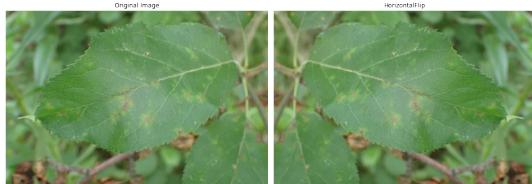


Fig. 12: RandomHorizontalFlip



Fig. 13: RandomBrightnessContrast

A. AutoAugment

除了图片展示的几种常见数据增广方法，我还使用了基于 AutoAugment 的数据增强方法：Learning Augmentation Strategies from Data。

通过创建一个搜索空间 (search space)，利用搜索算法 (search algorithm) 来选择合适的数据增强方法。

该方法在不同的数据集上具有良好的可迁移性 (transferable between datasets)，在一种数据集上学习到的方法 (policy) 迁移到其他数据集上也有不错的表现。

B. CutMix

之前的数据增强方法存在的问题：

- 1) mixup：混合后的图像在局部是模糊和不自然的，因此会混淆模型，尤其是在定位方面。
- 2) cutout：被 cutout 的部分通常用 0 或者随机噪声填充，这就导致在训练过程中这部分的信息被浪费掉了。

cutmix 在 cutout 的基础上进行改进，cutout 的部分用另一张图像上 cutout 的部分进行填充，这样即保留了 cutout 的优点：让模型从目标的部分视图去学习



Fig. 14: ShiftScaleRotate

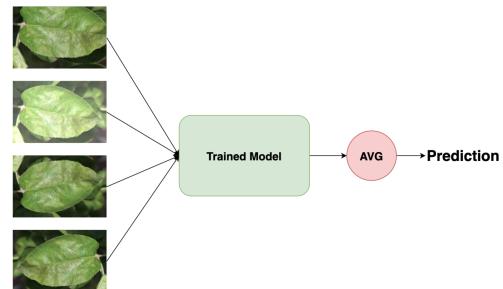


Fig. 15: Test Time Augmentation

目标的特征，让模型更关注那些 less discriminative 的部分。同时比 cutout 更高效，cutout 的部分用另一张图像的部分进行填充，让模型同时学习两个目标的特征。

cutmix 的具体过程如下

$$\begin{aligned}\tilde{x} &= \mathbf{M} \odot x_A + (\mathbf{1} - \mathbf{M}) \odot x_B \\ \tilde{y} &= \lambda y_A + (1 - \lambda) y_B,\end{aligned}$$

其中 $M \in \{0, 1\}^{W \times H}$ 是一个 binary mask 表明从两张图中裁剪的 patch 的位置，和 mixup 一样， λ 也是通过 $\beta(\alpha, \alpha)$ 分布得到的，在文章中作者设置 $\alpha = 1$ ，因此 λ 是，均匀分布 $(0, 1)$ 中采样的。为了得到 mask，首先要确定 cutmix 的 bounding box 的坐标 $B = (r_x, r_y, r_w, r_h)$ ，其值通过下式得到

$$\begin{aligned}r_x &\sim \text{Unif}(0, W), r_w = W\sqrt{1 - \lambda}, \\ r_y &\sim \text{Unif}(0, H), r_h = H\sqrt{1 - \lambda}\end{aligned}$$

即 λ 确定了 patch 与原图的面积比，即 A 图 cutout 的面积越大，标签融合时 A 图的比例越小。

C. TTA

TTA 又称为测试时数据增强 (test-time augmentation)，在测试时对图像进行数据增强，然后对增强后的图像进行预测，最后对预测结果进行平均，这样可以提高模型的泛化能力。

在这部分里我使用的比较特殊的操作是 Fixing Resolution。

FixRes (Fixing Resolution) 是一种能够提升任何模型性能的方法，可以在数个 epoch 期间用作卷积训练后的微调步骤，因而具有非常高的灵活性。FixRes 也可以轻松地集成到现有任何训练 pipeline。其核心是训练的时候使用小分辨率，测试的时候使用大分辨率（训练分辨率的 1.15 倍），能够有效提升验证精度。

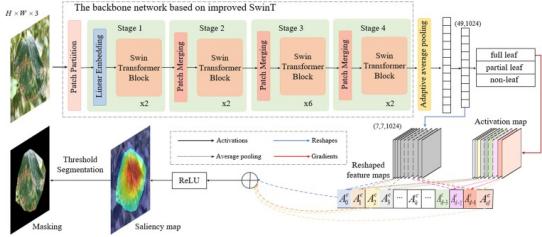


Fig. 16: GAN

D. GAN

LeafGAN 是一种有效的实用植物病害诊断数据扩充方法，它有自己的注意机制。LeafGAN 通过健康图像的转换生成各种各样的疾病图像，作为提高植物疾病诊断性能的数据增强工具。由于其自身的注意机制，模型只能从具有各种背景的图像中转换相关区域，从而丰富了训练图像的通用性。

在查找资料的过程中，我检索到一篇 2022 年 8 月的论文，其中将提出的主干网络和 Grad-CAM 组成的叶子提取模块集成到生成对抗网络（GAN）中，构建了 STA-GAN（基于 SwinT 和注意力引导的 GAN），仅在叶子区域生成病斑具有复杂背景的健康图像，用于增强疾病数据集。从实验结果来看，STA-GAN 表现出比 LeafGAN 更强的生成高质量图像的能力，甚至仅在 LeafGAN 消耗更多的训练图像时才近似。可惜代码还没开源，我也没能找到相关的开源代码，所以我就没有使用这个方法。

除了以上提及的方法之外，我还在 fgvc7 和 fgvc8 数据上进行了训练，提高了模型的泛化能力，fgvc7 需要将标签“multiple_diseases”替换为“complex”。

IX. 训练技巧

做完上述调研与准备之后我正式开始了训练，训练集与验证集合并后一共 3600 张图片，但我剔除了其中 healthy 标签的图片。故实际上模型标签有五类，在推理过程中，对所有疾病都呈阴性的图像被认为是健康的。

A. 交叉验证

在训练部分，我使用了 K-Fold Cross Validation ($k=5$)

交叉验证是一种通过将数据分为两部分来评估和比较学习算法的统计方法：一部分用于学习或训练模型，另一部分用于验证模型。在典型的交叉验证中，训

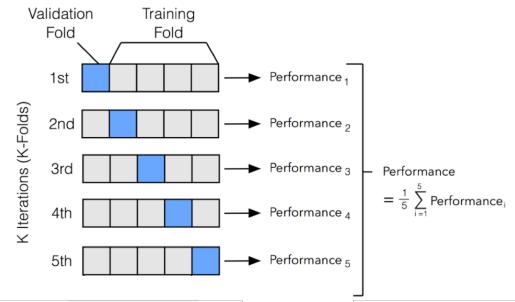


Fig. 17: 五折交叉验证

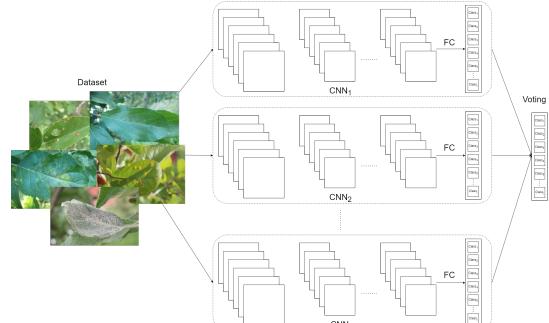


Fig. 18: 模型融合

练集和验证集必须在连续轮次中交叉，以便每个数据点都有机会被验证。我使用的是五折交叉验证，数据首先被划分为五份相同（或几乎相同）大小的“折”。随后执行训练和验证的迭代，以便在每次迭代中保留不同的数据“折”以进行验证，而剩余的 4 “折”用于学习。

B. 模型融合

模型融合是一种将多个模型的预测结果结合起来的方法，以提高预测准确性。模型融合的方法有很多，比如简单平均、加权平均、投票法、stacking、bagging、boosting 等。

当进行交叉验证时，在这个过程中，我产生了 K 个类似的模型，每个模型都是在有点不同的数据上训练的。因此，在每一折中，我都会产生一个不同的模型。我可以使用这些模型中的每一个，生成一个集合分类器，如上所述，这比使用单一模型要好。

C. 软化标签

由于疾病之间有些较难被区别，从而导致标签中存在一些不准确的情况，这给训练增加了一定的难度，我的模型很可能被这些不准确的标签给误导，为了应对这种情况的出现，我采用了自蒸馏的方式来解决该问

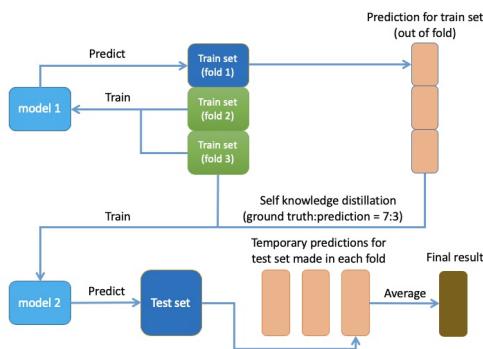


Fig. 19: 软化标签

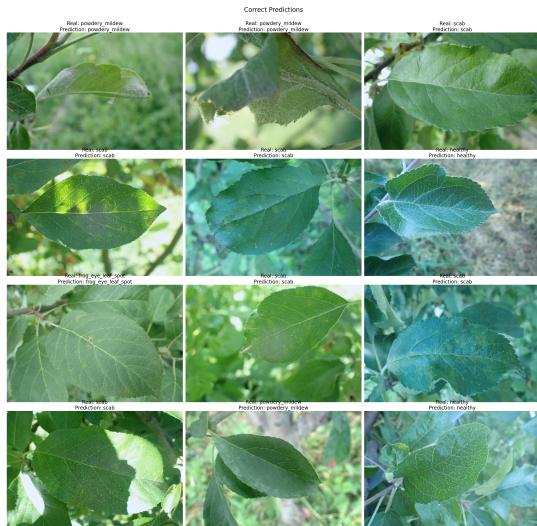


Fig. 20: 正确分类的图片

题，我训练了五折模型，然后将五折的验证集组成 out-of-fold 文件，最后将 out-of-fold 的结果和 groundtruth label 按 3: 7 混合作为训练新模型的标签，简单的描述即为给每个软化了之前的标签，给与每个标签一定的概率，进而减少了模型训练的难度。

X. 实验结果

综上所述，最终我的 best.ckpt 模型在 test 集的 600 张图片上的准确率为 90.5%，为了更直观的展示模型的效果，我做了以下几方面的可视化：

误差分析是迭代深度学习中十分重要的一个环节，当训练的模型完成后，如何对模型的性能进行改进才是提分的关键点，我通过对热力图的方式将模型对标签关键的识别部位提取出来，这样就能很清晰的知道模型主要看到了哪些部位才将图片识别为对应的类型，当我把识别错误的图片拿出来分析后，就可以知道数据增强的改进点和网络训练的改进点。（热力图除了使用 github

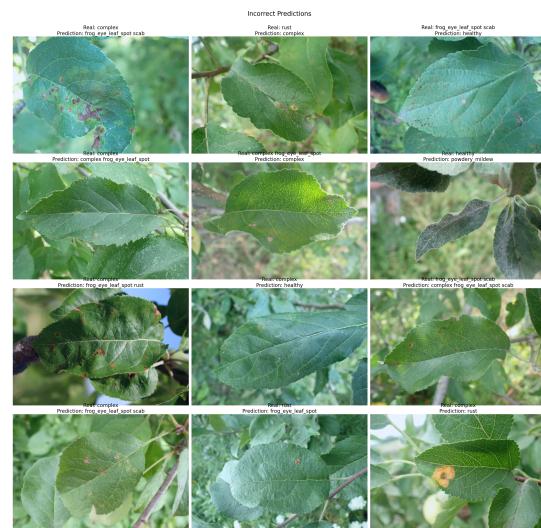


Fig. 21: 错误分类的图片

上的 pytorch-grad-cam 库外，也可以使用 mindspore 生态的 MindInsight 可视化调试调优工具）

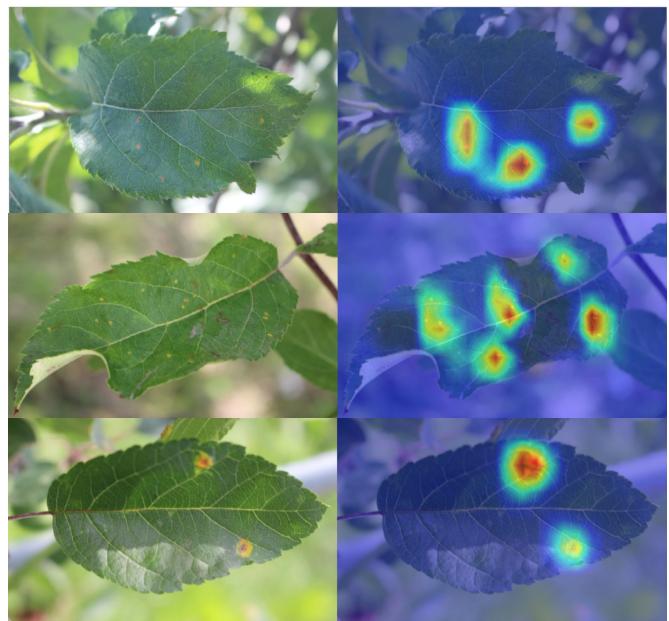


Fig. 22: 误差分析热力图

XI. 总结

本次作业中，我使用了 seresnext50_32x4d 作为 backbone，使用了多种数据增强方式，使用了五折交叉验证，模型融合和软化标签等技巧，最终在 test 集上的准确率为 90.5%。通过实现这些的学习还帮助我通过了华为的计算机视觉综合考核，希望在未来的学习中，能够更加深入地了解 MindSpore 生态。