

摘要： 对目前的一、二次采集的图像进行混合去训练

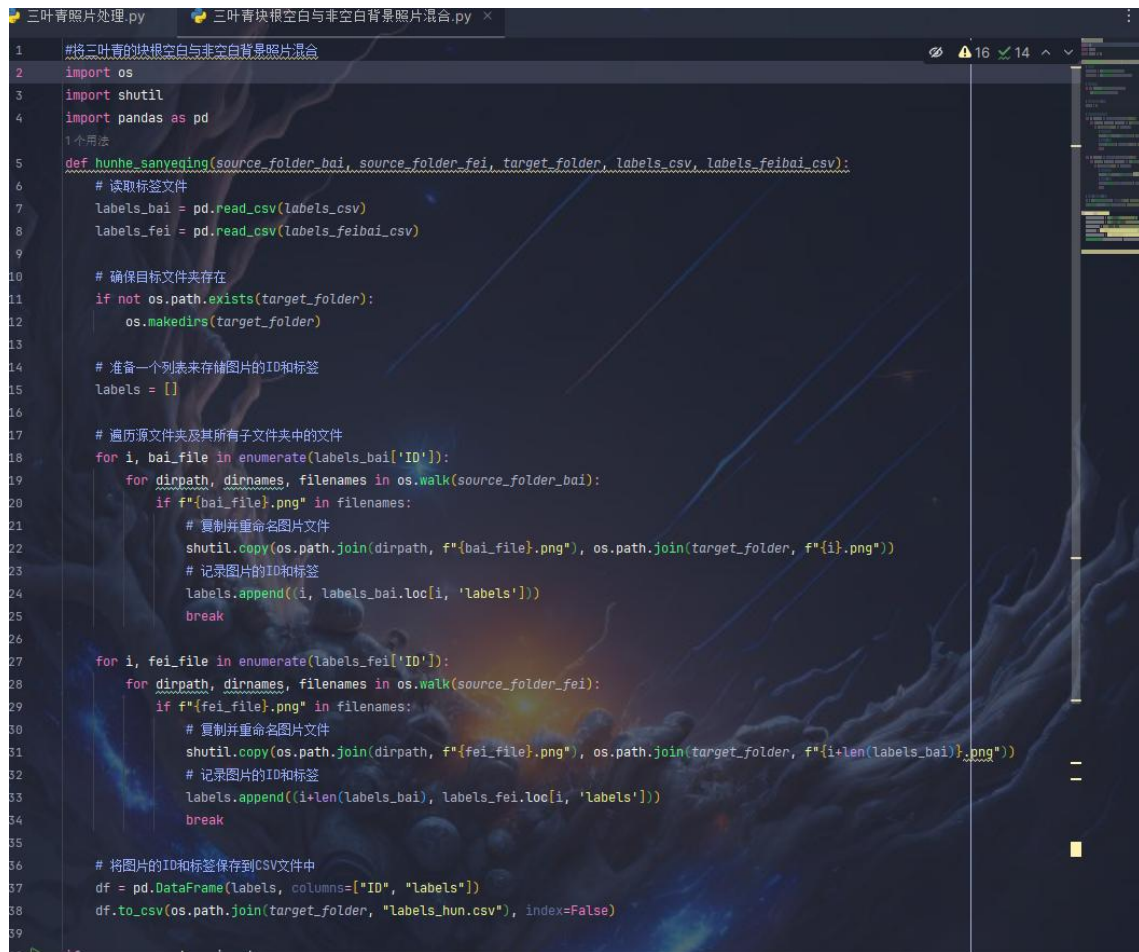
1. 实验准备

对三叶青图像进行二次采集，将第二次拍摄的照片与第一次的混合

2. 实验方法

2.1. 使用 python 脚本将照片混合

先用之前的处理三叶青的 python 脚本对第二次的三叶青图像处理后，再使用 python 脚本（下图所示）将第一二次处理好的三叶青图像混合



```
1 #将三叶青的块根空白与非空白背景照片混合
2 import os
3 import shutil
4 import pandas as pd
5 1个用法
6 def hunhe_sanveying(source_folder_bai, source_folder_fei, target_folder, labels_csv, labels_feibai_csv):
7     # 读取标签文件
8     labels_bai = pd.read_csv(labels_csv)
9     labels_fei = pd.read_csv(labels_feibai_csv)
10
11     # 确保目标文件夹存在
12     if not os.path.exists(target_folder):
13         os.makedirs(target_folder)
14
15     # 准备一个列表来存储图片的ID和标签
16     labels = []
17
18     # 遍历源文件夹及其所有子文件夹中的文件
19     for i, bai_file in enumerate(labels_bai['ID']):
20         for dirpath, dirnames, filenames in os.walk(source_folder_bai):
21             if f"{bai_file}.png" in filenames:
22                 # 复制并重命名图片文件
23                 shutil.copy(os.path.join(dirpath, f"{bai_file}.png"), os.path.join(target_folder, f"{i}.png"))
24                 # 记录图片的ID和标签
25                 labels.append((i, labels_bai.loc[i, 'Labels']))
26                 break
27
28     for i, fei_file in enumerate(labels_fei['ID']):
29         for dirpath, dirnames, filenames in os.walk(source_folder_fei):
30             if f"{fei_file}.png" in filenames:
31                 # 复制并重命名图片文件
32                 shutil.copy(os.path.join(dirpath, f"{fei_file}.png"), os.path.join(target_folder, f"{i+len(labels_bai)}.png"))
33                 # 记录图片的ID和标签
34                 labels.append((i+len(labels_bai), labels_fei.loc[i, 'Labels']))
35                 break
36
37     # 将图片的ID和标签保存到CSV文件中
38     df = pd.DataFrame(labels, columns=["ID", "Labels"])
39     df.to_csv(os.path.join(target_folder, "Labels_hun.csv"), index=False)
```

2.2. 在 kaggle 中训练

采用 resnet18 模型训练混合图像数据

```
# 使用示例
if __name__ == '__main__':
    # 定义要调整的参数范围
    param_grid = {
        'num_epochs': [50, 70, 90],
        'lr': [1e-4],
        'wd': [1e-4],
        'lr_period': [2],
        'lr_decay': [0.9]
    }

    print("这是十分类：")

    leibie_class("labels_hun.csv",
                num_classes=10,
                batch_size=128,
                valid_ratio=0.3,
                param_grid=param_grid,
                model_path="/kaggle/working/model_wht_shi.pth",
                target_dir="/kaggle/working/my_directory_shi",
                model_name='resnet18',
                model_leibie='get_resnet',
                train_folder='train_hun',
                test_folder='test_hun',
                model_path_zheng="/kaggle/working/model_wht_shi_zheng.pth") #十分类

train loss 0.341,train acc 0.878, valid acc 0.919,valid loss 0.244
1759.0 examples/sec on [device(type='cuda', index=0)]
Trial 1:
train loss 0.387,train acc 0.867, valid acc 0.911,valid loss 0.260
Trial 2:
train loss 0.365,train acc 0.867, valid acc 0.916,valid loss 0.249
Trial 3:
train loss 0.341,train acc 0.878, valid acc 0.919,valid loss 0.244
=====
Best accuracy: 0.919
Best params: {'num_epochs': 90, 'lr': 0.0001, 'wd': 0.0001, 'lr_period': 2, 'lr_decay': 0.9}
```

3. 实验结果

结果显示在 num_epochs=90 时，模型在训练集、验证集上的准确率比等于 50、70 的时候要高。说明了之后要再增加 num_epochs 数，或许模型效果更好。