机器学习课程设计报告：黄斑中心凹预测

徐浩淞20226446
1593120349@qq.com

摘要

本课题的目的是预测黄斑中央凹在图像中的坐标值。若图像中黄斑中央凹不可见，坐标值设为(0, 0)，否则需预测出中央凹在图像中的坐标值。

关键词： Macular Fovea Prediction ； Coordinate Regression ； Image Analysis

XuHaosong
1593120349@qq.com

Abstract

The purpose of this project is to predict the coordinate values of the macular fovea in the image. If the macular fovea is not visible in the image, set the coordinate value to (0,0), otherwise the coordinate value of the fovea in the image needs to be predicted.

Keywords: Macular Fovea Prediction , Coordinate Regression , Image Analysis

1 实验内容与具体要求

本课题的目的是预测黄斑中央凹在图像中的坐标值。若图像中黄斑中央凹不可见，坐标值设为(0, 0)，否则需预测出中央凹在图像中的坐标值。具体来说，为输出中心凹的坐标值，一种常用的策略是：首先需要完成黄斑区域的检测和定位；在该任务的基础上输出中心凹坐标值。因此，黄斑区域检测任务的性能影响中心凹的位置预测。由于没有对坐标(X,Y)的评价指标，因此提交文件的格式由坐标X/Y拆分为相邻的两行，再采用MSE作为最终的评价指标（MSE越小，排名越高），具体提交格式参照sample submission文件。

2 实验过程

2.1 定义标注转换成yolo格式的函数

用针对于xml的elementree方法，解析xml文件，并找到标注框节点，归一化中心x、y坐标、图像宽度、高度。

```python
def convert_xml_to_yolo(xml_path, img_width, img_height):#标注转换为yolo格式
    tree = ET.parse(xml_path)
    root = tree.getroot()

    bbox = root.find('.//bndbox')#标注框
    xmin = float(bbox.find('xmin').text)
    ymin = float(bbox.find('ymin').text)
    xmax = float(bbox.find('xmax').text)
    ymax = float(bbox.find('ymax').text)

    x_center = ((xmin + xmax) / 2) / img_width
    y_center = ((ymin + ymax) / 2) / img_height
    width = (xmax - xmin) / img_width
    height = (ymax - ymin) / img_height

    return [0, x_center, y_center, width, height]#0表示类别
```

Figure 1: 定义标注转换成yolo格式的函数

2.2 配置yolov8指定的数据集配置文件

根据yolo模型的需要，创建dataset,将数据集中的图片保存在dataset/imgaes/train中，以文本形式将标注框信息保存在dataset/labels/train中，并创建yaml文件，定义数据集路径、训练集、验证集、类别数、类别名称。

```python
def prepare_dataset():
    current_dir = os.path.dirname(os.path.abspath(__file__))

    dataset_dir = os.path.join(current_dir, 'dataset')#yolo规定数据集目录结构
    label_dir = os.path.join(dataset_dir, 'labels', 'train')
    train_dir = os.path.join(dataset_dir, 'images', 'train')
    os.makedirs(label_dir, exist_ok=True)
    os.makedirs(train_dir, exist_ok=True)

    xml_dir = os.path.join(current_dir, 'data', 'detection', 'train_location')
    img_dir = os.path.join(current_dir, 'data', 'detection', 'train')

    for xml_file in os.listdir(xml_dir):
        xml_path = os.path.join(xml_dir, xml_file)
        tree = ET.parse(xml_path)
        root = tree.getroot()

        size = root.find('size')
        width = int(size.find('width').text)
        height = int(size.find('height').text)
        yolo_bbox = convert_xml_to_yolo(xml_path, width, height)

        #保存标注
        img_id = xml_file.split('.')[0]
        label_path = os.path.join(label_dir, f'{img_id}.txt')
        with open(label_path, 'w') as f:
            f.write(' '.join(map(str, yolo_bbox)))#每行一个标注框信息

        img_file = f'{img_id}.jpg'
        detection_img_path = os.path.join(img_dir, img_file)
        dataset_img_path = os.path.join(train_dir, img_file)

        if os.path.exists(detection_img_path):
            shutil.copy2(detection_img_path, dataset_img_path)#把原来给出的训练集图像转到yolo指定的数据集目录

    dataset_yaml = os.path.join(dataset_dir, 'dataset.yaml')#yolo规定数据集配置文件
    with open(dataset_yaml, 'w') as f:
        f.write(
            f'''
            path: {dataset_dir}
            train: images/train
            val: images/train
            nc: 1#只有一个类别
            names: ['Fovea']
            '''
        )
```

Figure 2: 配置yolov8指定的数据集配置文件

2.3 设置参数，训练模型

使用os的文件处理方法，创建模型和yolo项目目录，直接调用yolo()加载模型，如果本地没有模型会自动下载，这里使用的是yolov8n模型，训练参数的设置是100轮，图像大小根据查询到的yolo最优参数设置为640，批量大小16，优化器使用adamw,学习率0.001。

```python
def train():
    current_dir = os.path.dirname(os.path.abspath(__file__))

    dataset_yaml = prepare_dataset()

    weights_dir = os.path.join(current_dir, 'weights')#模型
    os.makedirs(weights_dir, exist_ok=True)

    project_dir = os.path.join(current_dir, 'output')#结果
    os.makedirs(project_dir, exist_ok=True)

    model_name = 'yolov8n.pt'
    pretrained_model = os.path.join(weights_dir, model_name)

    # 加载模型
    if os.path.exists(pretrained_model):
        print(f"加载本地预训练模型: {pretrained_model}")
        model = YOLO(pretrained_model)
    else:
        print(f"下载{model_name}模型中。")
        model = YOLO(model_name)#如果加载不成功会自动下载并加载模型
        shutil.move(model_name, pretrained_model)

    # 训练模型
    model.train(
        data=dataset_yaml,
        epochs=100,
        imgsz=640,              #图像尺寸
        batch=16,               #批量大小
        name='fovea_detection',
        project=project_dir,
        exist_ok=True,
        optimizer='AdamW',
        lr0=0.001,              #学习率
    )
```

Figure 3: 设置参数，训练模型

2.4 预测测试集图片，输出结果

先加载训练好的模型，然后直接传入单张测试集图片路径进行预测，并使用opencv的imread函数获取测试图片的宽和高，用来预测失败的时候直接用中心点当目标点。预测成功则调用yolo结果对象的标注框的函数，获取中心坐标的x和y值，存到一个dataframe对象，便于转换为题目要求的csv格式。

```python
def predict():
    current_dir = os.path.dirname(os.path.abspath(__file__))

    weights_path = os.path.join(current_dir, 'output',  'fovea_detection', 'weights', 'best.pt')
    if not os.path.exists(weights_path):
        raise FileNotFoundError(f"找不到模型:{weights_path}\n")
    model = YOLO(weights_path)#加载训练好的模型

    results = []
    test_dir = os.path.join(current_dir, 'data', 'detection', 'test')

    #处理测试图像
    for img_file in sorted(os.listdir(test_dir)):
        img_path = os.path.join(test_dir, img_file)
        pred = model(img_path)[0]#返回的是result对象列表

        img = cv2.imread(img_path)#返回(height, width, channels)
        height, width = img.shape[:2]

        # 获取预测框的中心点
        if len(pred.boxes) > 0:
            box = pred.boxes[0]#返回第一个预测框，可信度最高
            x_center = float(box.xywh[0][0])
            y_center = float(box.xywh[0][1])
        else:
            x_center = width / 2#预测失败用图像最中心点代替
            y_center = height / 2

        # 添加结果
        img_id = img_file.split('.')[0]
        img_id = str(int(img_id))
        results.extend([
            {'ImageID': f'{img_id}_Fovea_X', 'value': x_center},
            {'ImageID': f'{img_id}_Fovea_Y', 'value': y_center}
        ])

    # 保存结果
    submission_path = os.path.join(current_dir, 'submission.csv')
    df = pd.DataFrame(results)#用DataFrame对象存结果
    df.to_csv(submission_path, index=False)#保存为CSV文件
    print(f"预测结果已保存到: {submission_path}")
```

Figure 4: 预测测试集图片，输出结果

3 实验结果

3.1 测试结果

MCtask_1 > submission.csv > data

```
ImageID,value
81_Fovea_X,1691.0474853515625
81_Fovea_Y,1105.0382080078125
82_Fovea_X,1692.8486328125
82_Fovea_Y,1106.04931640625
83_Fovea_X,1490.298095703125
83_Fovea_Y,1040.9254150390625
84_Fovea_X,1235.402099609375
84_Fovea_Y,1117.484130859375
85_Fovea_X,1205.6827392578125
85_Fovea_Y,983.1514892578125
86_Fovea_X,1257.2684326171875
86_Fovea_Y,1125.853515625
87_Fovea_X,1467.8875732421875
87_Fovea_Y,1015.14404296875
88_Fovea_X,1256.3173828125
88_Fovea_Y,1119.28857421875
89_Fovea_X,1477.748779296875
89_Fovea_Y,1041.06494140625
90_Fovea_X,1672.9365234375
90_Fovea_Y,1094.3251953125
91_Fovea_X,1460.526123046875
91_Fovea_Y,1055.623291015625
92_Fovea_X,1701.842529296875
92_Fovea_Y,1108.612060546875
93_Fovea_X,1696.0238037109375
93_Fovea_Y,1161.24169921875
94_Fovea_X,1443.3203125
94_Fovea_Y,1028.7406005859375
95_Fovea_X,1670.576171875
95_Fovea_Y,1096.50244140625
96_Fovea_X,1702.289306640625
96_Fovea_Y,1074.790283203125
97_Fovea_X,1504.895751953125
97_Fovea_Y,1212.3150634765625
98_Fovea_X,1471.3818359375
98_Fovea_Y,1064.2119140625
99_Fovea_X,1462.804443359375
99_Fovea_Y,1032.9947509765625
100_Fovea_X,1687.349853515625
100_Fovea_Y,1094.936767578125
```

Figure 5: 测试结果

3.2 测试得分

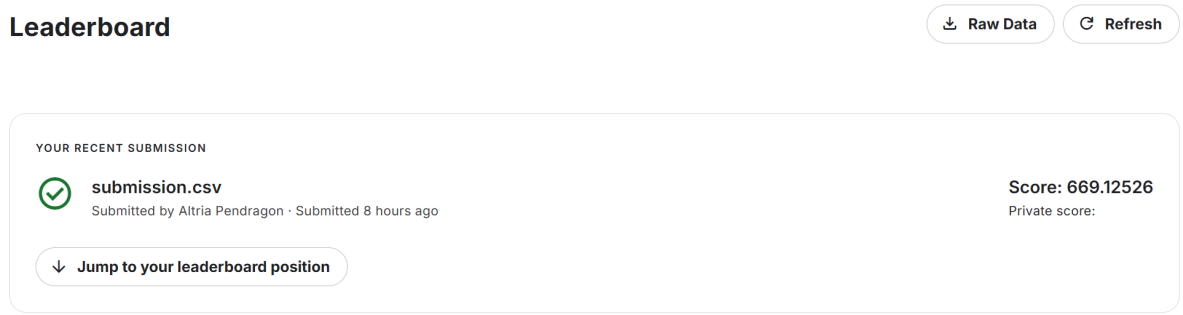

Figure 6: 测试得分

4 实验心得

通过课程设计的第一个任务，预测眼底黄斑中心凹，我学会了使用os方便快捷地创建文件目录、获取文件索引、提高了代码的通用性（不用担心换文件地址），还学习了如何使用预训练的yolov8n模型，如何将给出的标注数据和训练集、测试集图片转换到yolo指定的使用格式，

并且学习了配置yaml文件，用来训练yolo模型。还学习了如何预测图片，获取标注框信息，并且了解了可以用panda库创建一个dataframe对象存储列表，再转化为csv文件。