OpenCV Core AI

技术报告

项目名称：肺炎检测助手

团队名称：Pneumonia Detector

团队成员：李鑫、仲浩、李欣竹、王籽予

指导老师：贲晛烨、陈雷

完成时间：2022.12.2

目录

1 成员信息

姓名	学校	邮箱	竞赛经历
李鑫	山东大学	2561551088@qq.com	美国大学生数学建模竞赛优异奖
仲浩	山东大学	gunderp@foxmail.com	美国大学生数学建模竞赛优异奖
李欣竹	山东大学	572464728@qq.com	无
王籽予	山东大学	2497549465@qq.com	无
陈雷	山东大学	leiguanglg@163.com	导师
贲晛烨	山东大学	benxianye@126.com	导师

2 项目说明

鉴于目前新冠大流行的国际现状，核酸检测作为最方便、最流行的检测手段，经常出现阳性被误诊为阴性的情况，进而加大了阻断疫情传播工作的难度。

对此，我们团队认为，肺部 X 光和 CT 影像分析可以作为一种新的新冠检测手段，辅助核酸检测。考虑到传统的以医院为基础的医生分析检测方法，难以满足大量待检肺部 CT 图像的情况，我们团队致力于开发一个基于 OpenCV 和深度学习的肺炎检测平台，利用图像处理算法和 C#语言，对肺部影像进行分析处理。该平台使用 C#开发，数据集来源于网络上开源的肺炎 X 光数据集和肺部病灶分割数据集。

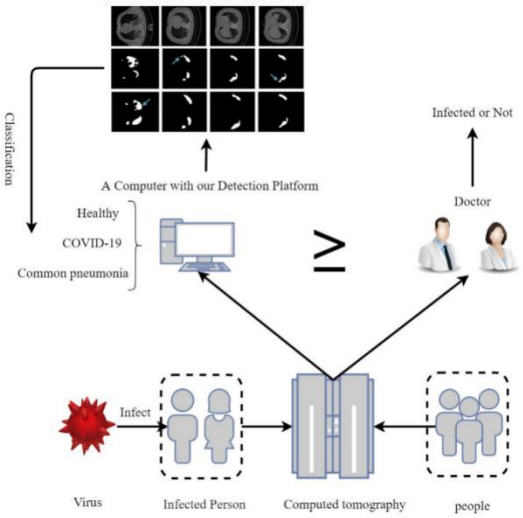
本项目通过图像识别和图像分割算法，代替传统的人工检测方式，最终实现了对正常肺部 X 光、病毒性肺炎 X 光、新冠肺炎 X 光进行分类和对肺部 CT 进行病灶分析，可以很好地作为医生诊断的辅助手段。

In view of the current status of the COVID- 19 , nucleic acid testing, as the most convenient and popular protocol, is often misdiagnosed as negative, thus making it more difficult to interrupt the transmission of the epidemic.

In this regard, our team believes that CT lung image analysis can be used as a

new detection method to assist in nucleic acid detection. Considering that the traditional hospital- based detection method of physician analysis is difficult to meet the large number of lung CT images to be examined, we are interested in developing a medical aid platform based on OpenCV by using image processing algorithms and C# language that can detect and identify lung CT images. The platform is developed using C#, and the dataset is derived from the open source dataset of New Coronary Pneumonia CT image segmentation on the web.

Through image recognition and image segmentation algorithms, instead of the traditional type of manual detection, normal lung CT, common pneumonia CT, and New Coronary Pneumonia CT can be processed and detected.



图一：项目流程图

3 项目原理与解决方案

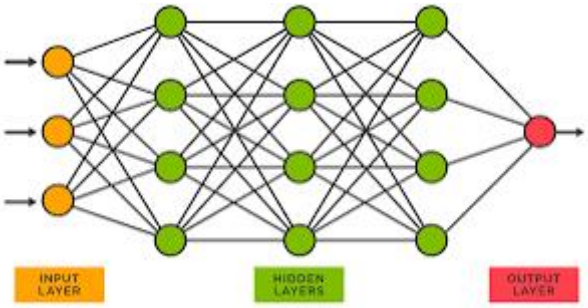
3.1 图像识别

本项目通过使用百度 PaddlePaddle 深度学习框架，基于网络上开源的肺炎 X 光影像数据集，搭建了计算机视觉中的经典模型 VGGNet，并通过 VGGNet 对肺炎 X 光影像进行识别。通过训练 VGGNet，神经网络在测试集上面最终得到了 96%的准确率，之后通过导出 ONNX 模型，并将其部署到 C#窗体程序中，最终实现了肺炎检测平台上的 X 光识别模块。

3.1.2 VGGNet 原理介绍：

3.1.2.1 神经网络:

人工神经网络是一个能够学习，能够总结归纳的系统，也就是说它能够通过已知数据的实验运用来学习和归纳总结。人工神经网络通过对局部情况的对照比较（而这些比较是基于不同情况下的自动学习和要实际解决问题的复杂性所决定的），它能够推理产生一个可以自动识别的系统。[1]



图二：人工神经网络

3.1.2.2 VGGNet：

VGGNet 是由牛津大学的计算机视觉小组在 2014 年 ILSVRC 中提出来的网络结构，在 2014 年的 ILSVRC 比赛中，VGG 在 Top-5 中取得了 92.3%的正确率，其提出的许多网络设计理念对后续神经网络的结构变迁产生了极大的影像，并且仍然活跃于许多迁移任务中。

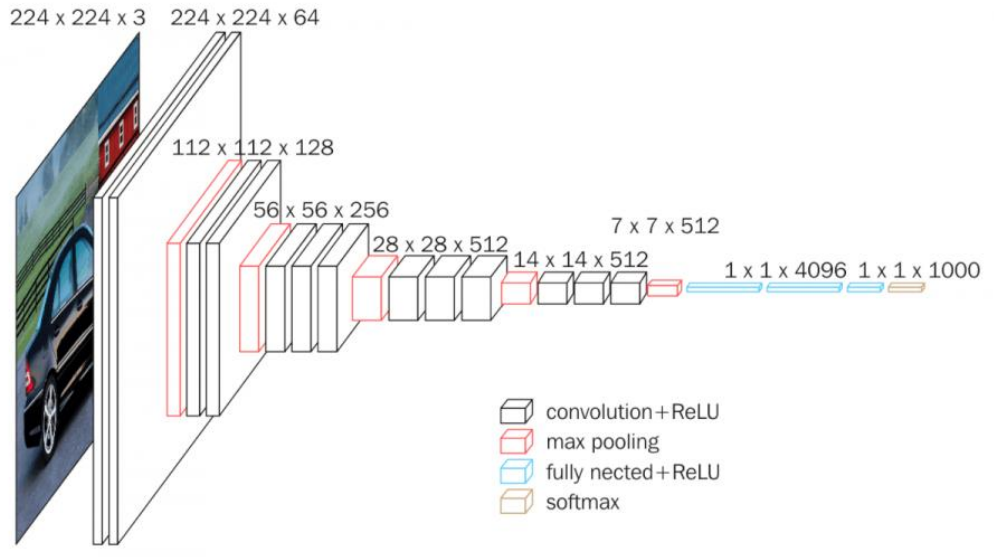
论文 Very Deep Convolutional Networks For Large-Scale Image Recognition[2] 主要提出了以下几点创新：

使用非常小的 3×3 卷积核来增加网络的深度进而对网络的性能进行研究，基于此方法可以将网络的深度推广到 16-19 层。[2]

3×3 的卷积核可以通过堆叠增加其感受野，两个 3×3 等同于一个 5×5 的感受野，三个等同于一个 7×7 卷积核的感受野。多个小卷积核的好处就是相对于一

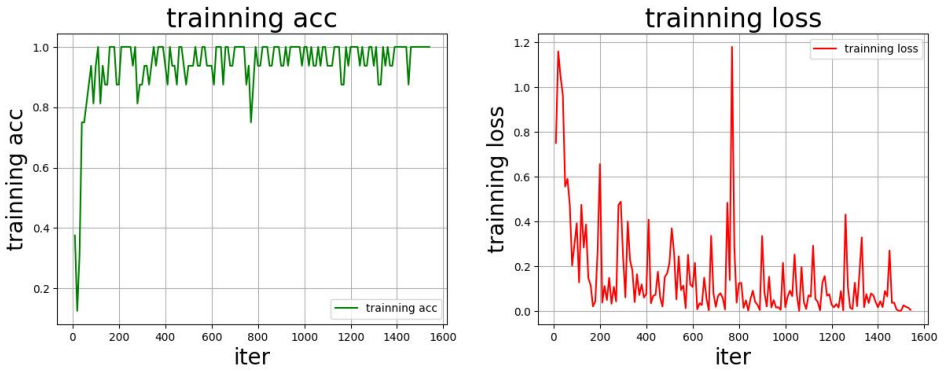
个大卷积核在其后面增加的多个非线性激活函数可以增强网络的非线性，同时也可以降低参数。[2]

1×1 卷积核的使用：增加非线性，1×1 卷积核本质上是对输入做了一个线性变换，之后经过了非线性的激活函数，因而增加了网络的非线性。[2]



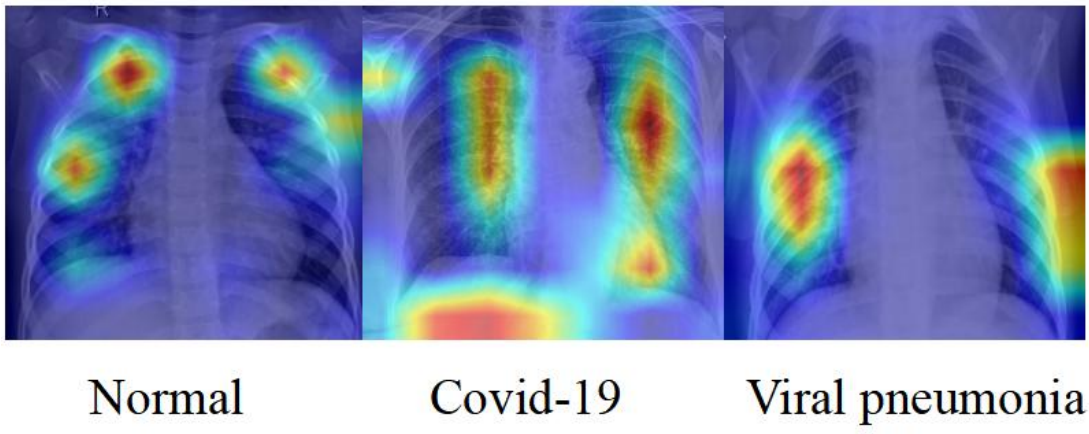
图三：VGGNet 网络结构图

在本项目中，我们对比使用了 VGGNet、ResNet18[3]、XceptionNet[4]以及 2020 年爆火的 ViT[5]模型，发现 VGG 在该数据集上表现最佳，ResNet18 的效果略差于 VGGNet，XceptionNet 由于轻量化设计，减少了许多参数，但是对应地，精度也有所下降，ViT 的效果只能达到 92%左右，从实验结果上也证明了 ViT 对比卷积模型拥有更小的归纳偏置[5]，在小数据集上的表现不佳。



图四：VGGNet 训练监视图

我们通过使用 Grad-CAM 方法，获得了 VGGNet 下的类别激活图可以看出网络对于不同的肺炎种类的关注区域差距较大。



图五：类别激活图

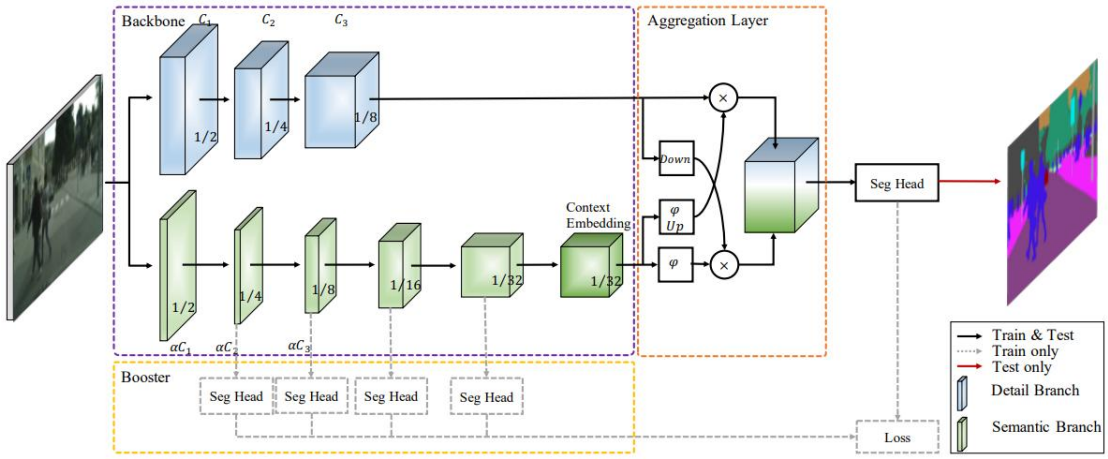
3.2 病灶分割

CT 影像技术在医学影像检测、分割、分类等诊断任务扮演着至关重要的角色，同时对于医生的诊断提供了证明与依据。尤其是在新型冠状病毒肺炎（COVID-19）近三年的流行期间，针对 CT 影像的深度学习图像处理与分析，很大程度上对于医生的决策起着不可忽视的辅助作用。因此能实现 CT 影像中的病灶分割，对医生的精确诊断与病人的尽早治疗提供更加高效的途径。但是新冠肺炎的病灶分割存在着一定的挑战，因为其早期呈现为磨玻璃样，边界与周围的组织器官边界模糊，存在结节状、斑片状或片状磨玻璃样密度影，病灶可相对局限或弥漫。[6]

本项目主要针对新冠肺炎的肺部 CT，使用 PaddlePaddle 深度学习框架，基于开源的新冠肺炎 CT 影像数据集，考虑项目的通常部署场景，搭建了计算机视觉中的轻量模型 BiSeNetV2，并通过 BiSeNetV2 对新冠肺炎 CT 影像中的病灶部分进行精准分割，并且在数据预处理与后处理阶段使用 OpenCV 计算机视觉库。最后采用了 PyQt 框架，实现了检测平台的开发与部署，但是考虑到医学影像数据一般是 Nifit 等三维数据，而训练过程中使用的是 2D 数据。因此在对影像数据进行推理时，是采用分层推理的方式，再堆叠结果进行层层展示，并在原图在病灶位置处绘制轮廓，最终实现了新冠肺炎检测平台上的病灶分割模块。

3.2.2 BiSeNet V2 原理介绍：

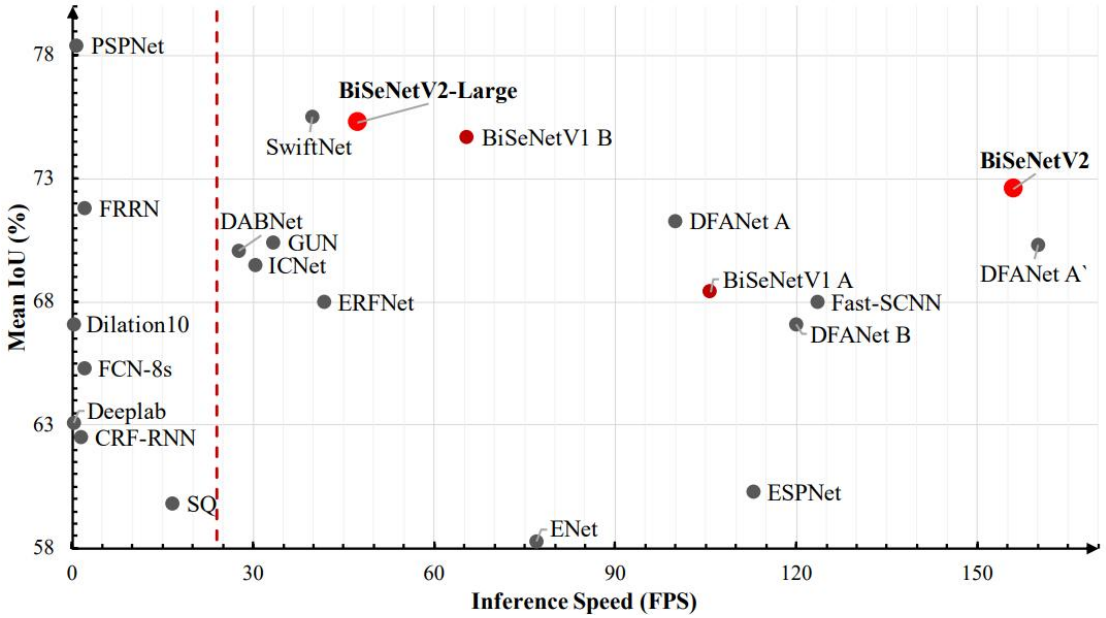
BiSeNetV2[7]是由华中科技大学计算机视觉小组提出来的一种在速度和准确性之间进行良好权衡的双边分割网络（BiSeNet V2）。其核心思想是，浅层细节（low-level details）和深层语义（high-level semantics）对语义分割都非常重要，可以考虑将空间细节和类别语义分开处理，以实现精度和推理速度之间的平衡。模型的架构图如下：



图六：BiSeNet V2 网络结构图

其中 Detail Branch 的任务是保存低级特征的空间细节信息，所以该分支的设计理念就是通道多，层数少，下采样比率低。为了防止模型速度降低，没有采用 residual connection[3]。Semantic Branch 的目标是捕获高级的语义信息，所示他的设计理念是通道少，层数多，下采样比率大（增加感受野）。

架构设计概括为：细节分支（Detail Branch）被设计为宽通道、层数较浅，用于捕捉空间细节；语义分支（Semantic Branch）通道较窄、层数较深，用于提取类别语义。此外，作者设计了一个引导聚合层(Guided Aggregation Layer)，以增强双向连接并融合这两种类型的特征表示。还设计了一种增强训练策略来提高分割性能，而不需要任何额外的推理成本。



图七：Cityscapes 测试集下性能测试

与几种最新的实时语义分割方法相比，BiSeNet V2 具有良好的性能：模型的推理速度比现有方法快得多的同时，可以实现更好的分割精度。

我们在实现 CT 影像分割时，在 UNet，LOD-Net，BiSeNet V2 等模型中，综合考虑分割的精度与部署的场景，选择 BiSeNet V2 作为最终的系统模型。

4 代码结构

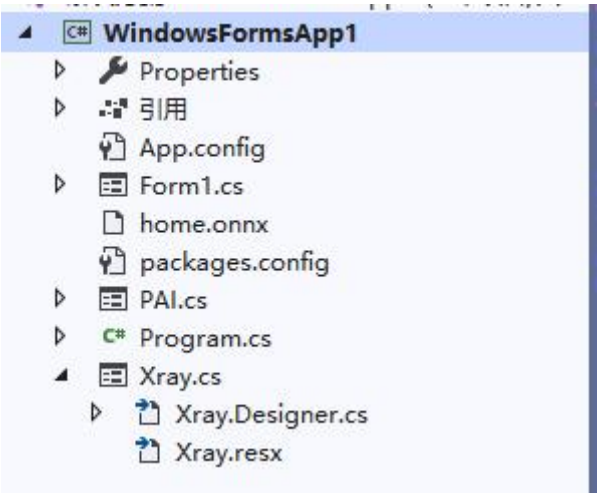
4.1 C#代码结构

Form1.cs 为程序主界面

home.onnx 为 VGGNet 导出的 onnx 文件

PAI.cs 为项目说明界面

Xray.cs 为 X 光识别界面



图八：C#代码结构

4.2 VGG 分类模型代码

4.2.1 环境配置

表一：分类模型环境要求

库名	版本
python	3.7
paddle	2.2.2
json5	0.9.5
matplotlib	2.2.3
numpy	1.21.6
Pillow	8.2.0

4.2.2 VGGNet 代码架构：

图九：VGGNet 分类代码架构

4.2.3 VGGNet 文件说明：

train.py:训练脚本

test.py:训练脚本

eval.py:评估脚本

model.py:定义网络结构

data_processing.py:解压数据、划分数据集

Dataset.py:继承 paddle 的 dataset,适配本数据集。

onnx_output.py:输出 onnx 模型

work/checkpoints：存放训练后参数

data/dataset:存放数据集的目录

4.3 BiSeNet V2 分割模型代码

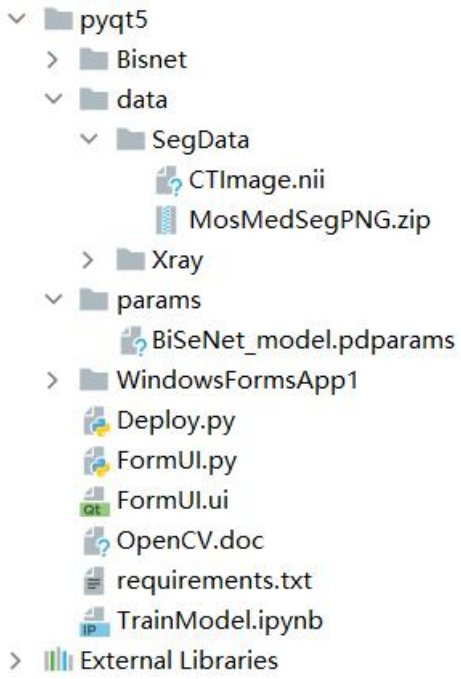
4.3.1 环境配置

表二：分割模型环境要求

库名	版本

python	3.7
numpy	1.21.2
opencv-python	4.5.3.56
paddlepaddle	2.1.3
paddleseg	2.3.0
PyQt5	5.15.6
SimpleITK	2.1.1

4.3.2 BiSeNet V2 代码架构：



图十：BiSeNet V2 分割代码架构

4.3.3 BiSeNet V2 文件说明：

CTImage：CT 影像文件

BiSeNet_model.pdparams：BiSeNet V2 模型的训练参数

Deploy.py：具体部署与系统的实现

FormUI.py：窗体的转换 py 文件

FormUI.ui：窗体的 UI 文件

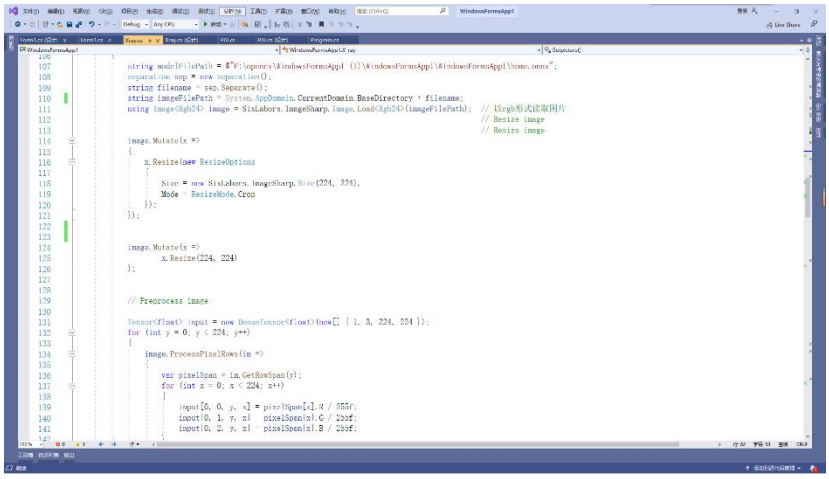
requirements.txt：项目所需的环境依赖包以及对应版本

TrainModel.ipynb：模型训练文件，采用 paddle 平台训练

MosMedSegPNG.zip：训练采用的数据集

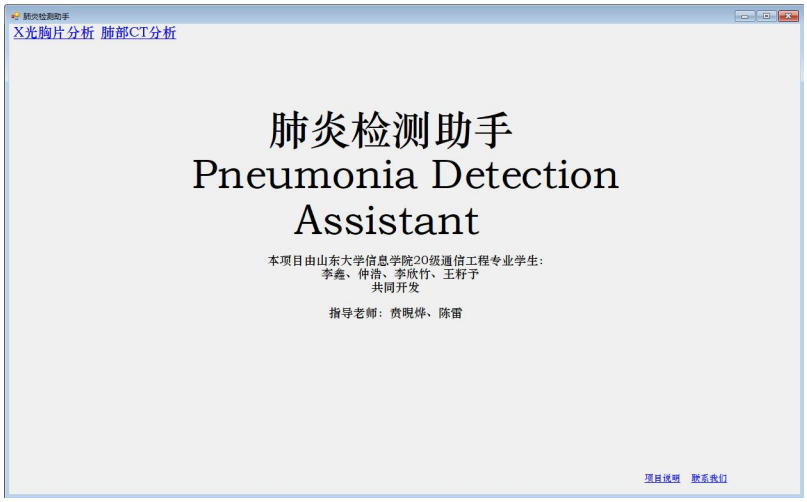
5 使用方法

使用 VS2019 打开文件夹下的 sln 文件



图十一：运行代码

点击运行（绿色三角按钮）



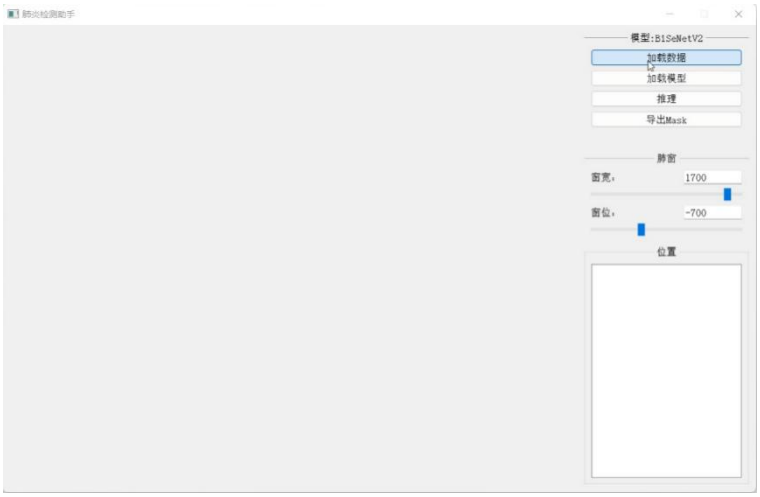
图十二：项目主页面

点击 X 光胸片分析，进入页面，点击图片上传，之后任选取一个数据集中的图片，点击 AI 分析，得到置信度和检测结果

图十三：X 光分类页面

点击返回，返回主页面

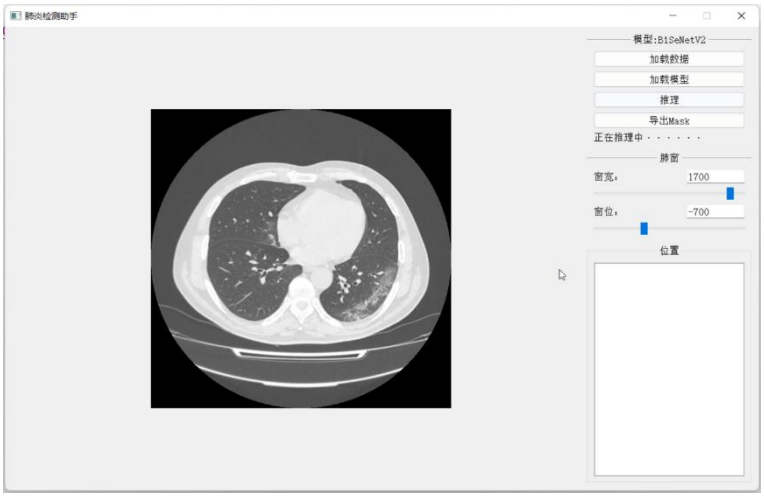
点击肺部 CT 分析



图十四：CT 分割初始页面

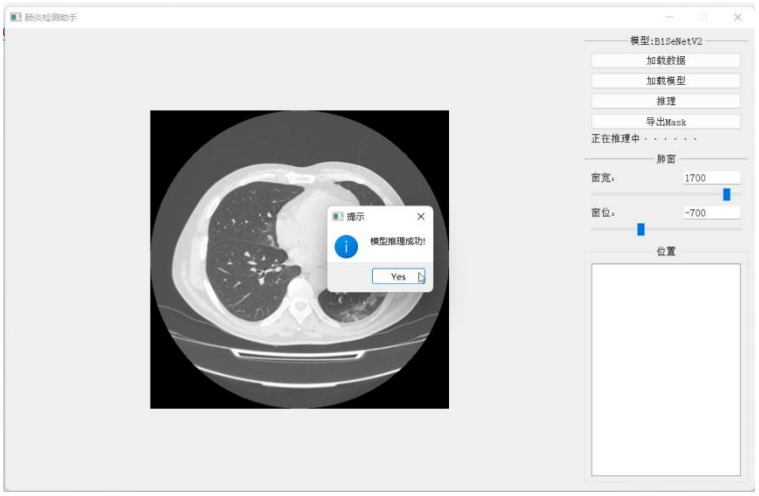
分别点击选择加载数据、加载模型和推理（加载的数据和加载模型的文件都放在 github 上）

显示模型正在推理：



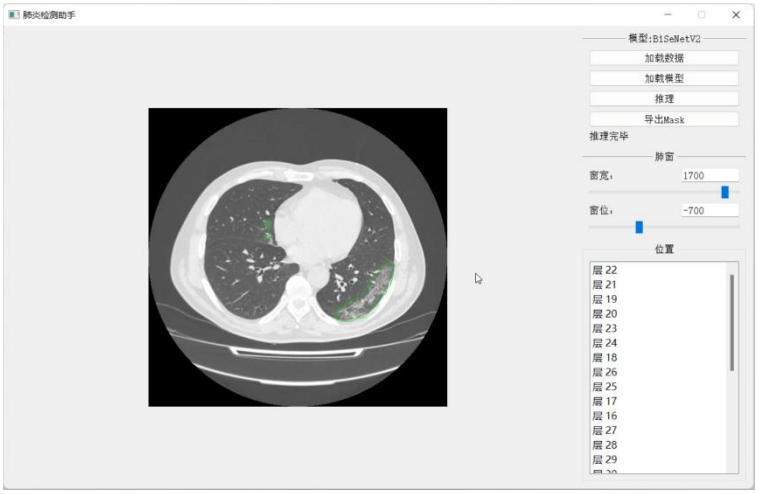
图十五：加载模型和加载数据

推理成功后会有弹窗提醒：



图十六：推理成功

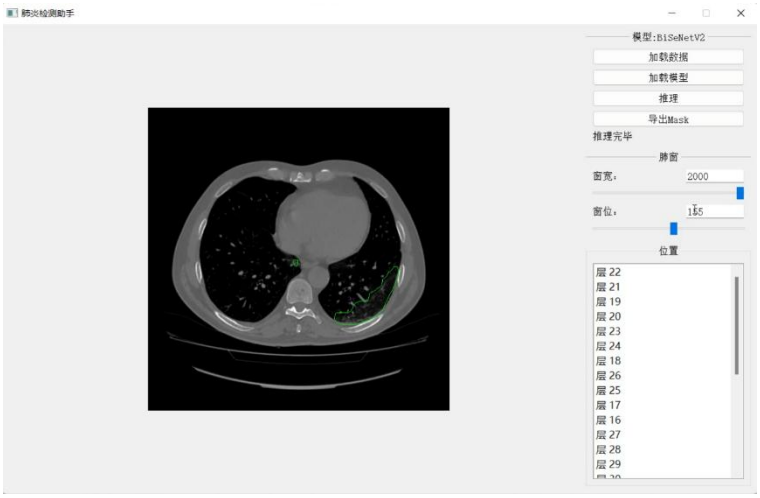
推理结果展示：



图十七：推理结果截图

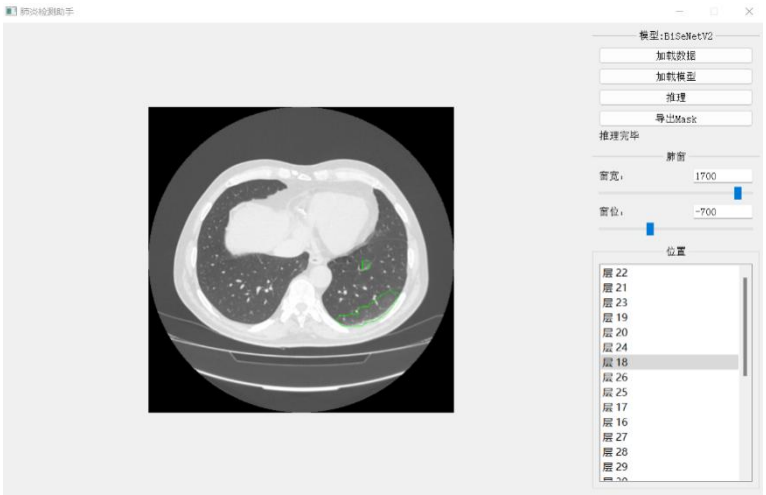
调节肺窗的窗宽和窗位的效果；

图十八：改变窗宽



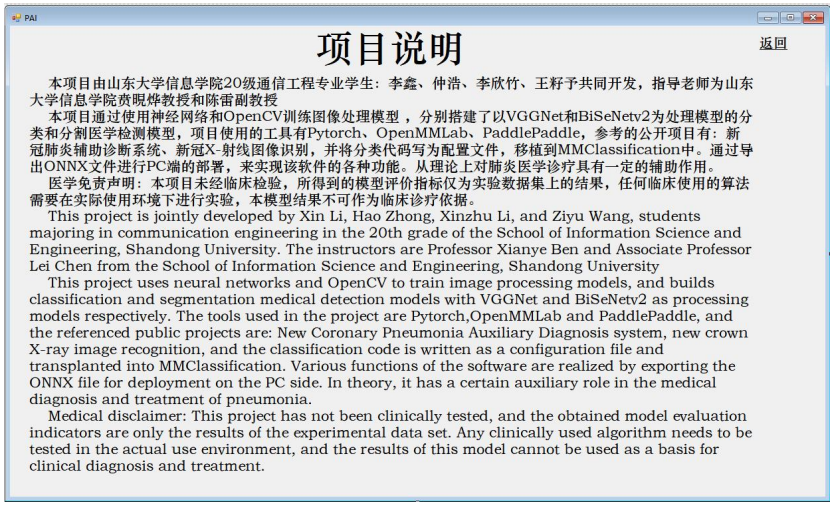
图十九：改变窗位

还可以通过位置层的选择，精准地还原三维图像中的病灶分布：



图二十：位置层选择

点击项目说明：



图二十一：项目说明页面

点击联系我们，会跳转到我们的代码开源仓库。

如果想移植我们的项目，你只需要修改以下路径即可：

1. Xray.cs 文件中的：onnx 文件的存放路径
2. Form1.cs 文件中的：process.StartInfo.Arguments，设置为分割模型 Deploy.py 的路径。

如果你有什么问题：

请在我们的 github 仓库中提 issue。

我们的 github 地址：https://github.com/Laymanpython/OpenCV_Competition

致　谢

感谢 OpenCV 基金会提供的本次比赛机会，通过本项目的开发，我们将自己在大学中所学到的专业知识应用在实际问题上，既巩固了自己的理论知识，又锻炼了工程能力，是一次非常宝贵的实训经历，为我们日后从事 AI 行业打下了坚实的基础。

感谢我们的导师贲晛烨教授和陈雷副教授，两位老师是我们在工程项目和学术研究上的引路人和榜样，从比赛开始的选题到项目结题，两位老师一直都在给予我们团队最大的帮助，在开发过程中给了许多的鼓励和指导，使我们能够顺利

完成该项目的开发，尤其是贲老师的 C#课程，将面向对象编程的思想融入一个个具体的实例，让我们对面向对象的各种语言有了更深一层的认知，在开发实践中心手相应。

感谢 OpenCV 的所有开发者，OpenCV 的诞生减轻了所有计算机视觉领域的研究者和从业者的压力，其优秀的技术文档和简易的代码风格使得 OpenCV 成为了很多计算机视觉初学者的入门工具，本次项目开发过程中我们也选择使用了 OpenCV 作为我们的开发工具，同时通过浏览 OpenCV 开源社区的相关帖子，得到了不少的帮助，也开拓了我们解决问题的思路。

感谢我们团队的每位成员，项目的完成离不开每个人的努力和付出，是项目进入低谷时的相互鼓励和对我们自己的信心支撑着我们解决了一个又一个的问题，直至完成项目。从选题确定到最后开发完成，我们克服了一个接一个的难关，完成了预期的所有功能，相信这段一起开发的经历，将会成为我们人生中最难忘的一段时光。

最后，衷心祝愿 AI 行业继续蓬勃发展，助力各项传统工业、服务业、制造业做好智能化的转型与落地，疫情阴霾早日消散，人民生活迎来曙光。

参 考 文 献

1.https://zh.m.wikipedia.org/zhhans/%E4%BA%BA%E5%B7%A5%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C

2. K. Simonyan and A. Zisserman. (2014). “Very deep convolutional networks for large-scale image recognition.” [Online]. Available: https://arxiv.org/abs/1409.1556.

3. He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

4. Chollet F. Xception: Deep learning with depthwise separable convolutions[J]. arXiv preprint, 2017: 1610.02357.

5. Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16x16 words: Transformers for image recognition at scale[J]. arXiv preprint arXiv:2010.11929,

2020.

6. Lei J,Li J,Li X,et al.CT Imaging of the 2019 Novel Coronavirus(2019-nCoV) Pneumonia[J].Radiology,2020,295(1):18

7. Yu, C., Gao, C., Wang, J. et al. BiSeNet V2: Bilateral Network with Guided Aggregation for Real-Time Semantic Segmentation. Int J Comput Vis 129, 3051–3068 (2021).

附录

分类模型代码：

data_processing.py

```
import os
import zipfile
import random
import json

'''
参数配置
'''
train_parameters = {
    "input_size": [3, 224, 224],  # 输入图片的 shape
    "class_dim": 3,  # 分类数
    "src_path": "data/dataset.zip",  # 原始数据集路径
    "target_path": "data/dataset",  # 要解压的路径
    "train_list_path": "data/train.txt",  # train.txt 路径
    "eval_list_path": "data/eval.txt",  # eval.txt 路径
    "readme_path": "data/readme.json",  # readme.json 路径
    "label_dict": {},  # 标签字典
    "num_epochs": 20,  # 训练轮数
    "train_batch_size": 64,  # 训练时每个批次的大小
    "skip_steps": 30,
    "save_steps": 300,
    "learning_strategy": {  # 优化函数相关的配置
        "lr": 0.0001  # 超参数学习率
    },
```

```
    "checkpoints": "work/checkpoints"  # 保存的路径

}


def unzip_data(src_path, target_path):
    '''
    解压原始数据集，将 src_path 路径下的 zip 包解压至 target_path 目录下
    '''
    if (not os.path.isdir(target_path + "Chinese Medicine")):
        z = zipfile.ZipFile(src_path, 'r')
        z.extractall(path=target_path)
        z.close()


def get_data_list(target_path, train_list_path, eval_list_path):
    '''
    生成数据列表
    '''
    # 存放所有类别的信息
    class_detail = []
    # 获取所有类别保存的文件夹名称
    data_list_path = target_path + "/train/"
    class_dirs = os.listdir(data_list_path)
    # 总的图像数量
    all_class_images = 0
    # 存放类别标签
    class_label = 0
    # 存放类别数目
    class_dim = 0
    # 存储要写进 eval.txt 和 train.txt 中的内容
    trainer_list = []
    eval_list = []
    # 读取每个类别
    for class_dir in class_dirs:
        if class_dir != ".DS_Store":
            class_dim += 1
            # 每个类别的信息
            class_detail_list = {}
            eval_sum = 0
            trainer_sum = 0
            # 统计每个类别有多少张图片
            class_sum = 0
            # 获取类别路径
```

```
69.             path = data_list_path + class_dir
70.             # 获取所有图片
71.             img_paths = os.listdir(path)
72.             for img_path in img_paths:  # 遍历文件夹下的每个图片
73.                 if img_path.split(".")[-1] == "png":
74.                     name_path = path + '/' + img_path  # 每张图片的路径
75.                     if class_sum % 8 == 0:  # 每8张图片取一个做验证数据
76.                         eval_sum += 1  # test_sum 为测试数据的数目
77.                         eval_list.append(name_path + "\t%d" % class_label + "\n")
78.                     else:
79.                         trainer_sum += 1
80.                         trainer_list.append(name_path + "\t%d" % class_label + "\n")  # trainer_sum 测试数据的数目
81.                     class_sum += 1  # 每类图片的数目
82.                     all_class_images += 1  # 所有类图片的数目
83.                 else:
84.                     continue
85.             # 说明的 json 文件的 class_detail 数据
86.             class_detail_list['class_name'] = class_dir  # 类别名称
87.             class_detail_list['class_label'] = class_label  # 类别标签
88.             class_detail_list['class_eval_images'] = eval_sum  # 该类数据的测试集数目
89.             class_detail_list['class_trainer_images'] = trainer_sum  # 该类数据的训练集数目
90.             class_detail.append(class_detail_list)
91.             # 初始化标签列表
92.             train_parameters['label_dict'][str(class_label)] = class_dir
93.             class_label += 1
94.
95.             # 初始化分类数
96.     train_parameters['class_dim'] = class_dim
97.
98.     # 乱序
99.     random.shuffle(eval_list)
100.     with open(eval_list_path, 'a') as f:
101.         for eval_image in eval_list:
102.             f.write(eval_image)
103.
```

```
    random.shuffle(trainer_list)
    with open(train_list_path, 'a') as f2:
        for train_image in trainer_list:
            f2.write(train_image)

            # 说明的 json 文件信息
    readjson = {}
    readjson['all_class_name'] = data_list_path  # 文件父目录
    readjson['all_class_images'] = all_class_images
    readjson['class_detail'] = class_detail
    jsons = json.dumps(readjson, sort_keys=True, indent=4, separators=(',', ': '))
    with open(train_parameters['readme_path'], 'w') as f:
        f.write(jsons)
    print('生成数据列表完成! ')

```

Dataset.py

```
import os
import zipfile
import random
import json
import paddle
import sys
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
from paddle.io import Dataset
from data_processing import  unzip_data,train_parameters

class dataset(Dataset):
    def __init__(self, data_path, mode='train'):
        """
        数据读取器
        :param data_path: 数据集所在路径
        :param mode: train or eval
        """
        super().__init__()
        self.data_path = data_path
        self.img_paths = []
        self.labels = []

        if mode == 'train':
```

```
            with open(os.path.join(self.data_path, "train.txt"), "r", encoding="utf-8") as f:
                self.info = f.readlines()
            for img_info in self.info:
                img_path, label = img_info.strip().split('\t')
                self.img_paths.append(img_path)
                self.labels.append(int(label))

        else:
            with open(os.path.join(self.data_path, "eval.txt"), "r", encoding="utf-8") as f:
                self.info = f.readlines()
            for img_info in self.info:
                img_path, label = img_info.strip().split('\t')
                self.img_paths.append(img_path)
                self.labels.append(int(label))


    def __getitem__(self, index):
        """
        获取一组数据
        :param index: 文件索引号
        :return:
        """
        # 第一步打开图像文件并获取 label 值
        img_path = self.img_paths[index]
        img = Image.open(img_path)
        if img.mode != 'RGB':
            img = img.convert('RGB')
        img = img.resize((224, 224), Image.BILINEAR)
        img = np.array(img).astype('float32')
        img = img.transpose((2, 0, 1)) / 255
        label = self.labels[index]
        label = np.array([label], dtype="int64")
        return img, label

    def print_sample(self, index: int = 0):
        print("文件名", self.img_paths[index], "\t 标签值", self.labels[index])

    def __len__(self):
        return len(self.img_paths)
```

eval.py

```
import paddle
import numpy as np
from Dataset import dataset
from model import VGGNet
'''
模型评估
'''
# 测试数据加载
eval_dataset = dataset('/home/aistudio/data', mode='eval')
eval_loader = paddle.io.DataLoader(eval_dataset, batch_size=8, shuffle=False)
eval_dataset.print_sample(0)
print(eval_dataset.__len__())
print(eval_dataset.__getitem__(10)[0].shape)
print(eval_dataset.__getitem__(10)[1].shape)
model__state_dict = paddle.load('work/checkpoints/save_dir_1500.pdparams')
model_eval = VGGNet()
model_eval.set_state_dict(model__state_dict)
model_eval.eval()
accs = []

for _, data in enumerate(eval_loader()):
    x_data = data[0]
    y_data = data[1]
    predicts = model_eval(x_data)
    acc = paddle.metric.accuracy(predicts, y_data)
    accs.append(acc.numpy()[0])
print('模型在验证集上的准确率为：',np.mean(accs))

```

paddle.py

```
import  paddle


class  ConvPool(paddle.nn.Layer):
        '''卷积+池化'''

        def  __init__(self,
                                num_channels,
                                num_filters,
                                filter_size,
```

```
                                    pool_size,
                                    pool_stride,
                                    groups,
                                    conv_stride=1,
                                    conv_padding=1,
                                    ):
                super(ConvPool,  self).__init__()

                for  i  in  range(groups):
                        conv2d  =  self.add_sublayer(    #  添加子层实例
                                'bb_%d'  %  i,
                                paddle.nn.Conv2D(    #  layer
                                        in_channels=num_channels,    #
通道数
                                        out_channels=num_filters,    #
卷积核个数
                                        kernel_size=filter_size,    #
卷积核大小
                                        stride=conv_stride,    #  步长
                                        padding=conv_padding,    #  padd
ing
                                )
                        )
                        self.add_sublayer(
                                'relu%d'  %  i,
                                paddle.nn.ReLU()
                        )
                        num_channels  =  num_filters

                self.add_sublayer(
                        'Maxpool',
                        paddle.nn.MaxPool2D(
                                kernel_size=pool_size,    #  池化核大小
                                stride=pool_stride    #  池化步长
                        )
                )

        def  forward(self,  inputs):
                x  =  inputs
                for  prefix,  sub_layer  in  self.named_children():
                        #  print(prefix,sub_layer)
                        x  =  sub_layer(x)
                return  x

```

```
51.
52. #  使用上面的 ConvPool 模块定义 VGGNet
53. class  VGGNet(paddle.nn.Layer):
54.
55.         def  __init__(self):
56.                 super(VGGNet,  self).__init__()
57.                 #  #3:通道数，64：卷积核个数，3:卷积核大小，2:池化核大小，2:池化步长，2:连续卷积个数(每两组之间)
58.                 self.convpool1  =  ConvPool(3,  64,  3,  2,  2,  2)
59.                 self.convpool2  =  ConvPool(64,  128,  3,  2,  2,  2)
60.                 self.convpool3  =  ConvPool(128,  256,  3,  2,  2,  3)
61.                 self.convpool4  =  ConvPool(256,  512,  3,  2,  2,  3)
62.                 self.convpool5  =  ConvPool(512,  512,  3,  2,  2,  3)
63.                 self.convpool5_shape  =  512  *  7  *  7
64.                 self.fc1  =  paddle.nn.Linear(self.convpool5_shape,  4096)
65.                 self.fc2  =  paddle.nn.Linear(4096,  4096)
66.                 self.fc3  =  paddle.nn.Linear(4096,  3)
67.
68.         def  forward(self,  inputs,  label=None):
69.
70.                 x  =  self.convpool1(inputs)
71.                 x  =  self.convpool2(x)
72.                 x  =  self.convpool3(x)
73.                 x  =  self.convpool4(x)
74.                 x  =  self.convpool5(x)
75.                 x  =  paddle.reshape(x,  [-1,  512  *  7  *  7])
76.                 x  =  self.fc1(x)
77.                 x  =  self.fc2(x)
78.                 out  =  self.fc3(x)
79.
80.                 if  label  is  not  None:
81.                         acc  =  paddle.metric.accuracy(input=out,  label=label)
82.                         return  out,  acc
83.                 else:
84.                         return  out
85.
86.
```

onnx_output.py

```
1. import paddle
```

```
2.
3.
4. class ConvPool(paddle.nn.Layer):
5.     '''卷积+池化'''
6.
7.     def __init__(self,
8.                  num_channels,
9.                  num_filters,
10.                 filter_size,
11.                 pool_size,
12.                 pool_stride,
13.                 groups,
14.                 conv_stride=1,
15.                 conv_padding=1,
16.                 ):
17.         super(ConvPool, self).__init__()
18.
19.         for i in range(groups):
20.             conv2d = self.add_sublayer(  # 添加子层实例
21.                 'bb_%d' % i,
22.                 paddle.nn.Conv2D(  # layer
23.                     in_channels=num_channels,  # 通道数
24.                     out_channels=num_filters,  # 卷积核个数
25.                     kernel_size=filter_size,  # 卷积核大小
26.                     stride=conv_stride,  # 步长
27.                     padding=conv_padding,  # padding
28.                 )
29.             )
30.             self.add_sublayer(
31.                 'relu%d' % i,
32.                 paddle.nn.ReLU()
33.             )
34.             num_channels = num_filters
35.
36.         self.add_sublayer(
37.             'Maxpool',
38.             paddle.nn.MaxPool2D(
39.                 kernel_size=pool_size,  # 池化核大小
40.                 stride=pool_stride  # 池化步长
41.             )
42.         )
43.
44.     def forward(self, inputs):
45.         x = inputs
```

```
        for prefix, sub_layer in self.named_children():
            x = sub_layer(x)
        return x


# 使用上面的 ConvPool 模块定义 VGGNet
class VGGNet(paddle.nn.Layer):

    def __init__(self):
        super(VGGNet, self).__init__()
        # #3:通道数，64：卷积核个数，3:卷积核大小，2:池化核大小，2:池化步长，2:连续卷积个数(每两组之间)
        self.convpool1 = ConvPool(3, 64, 3, 2, 2, 2)
        self.convpool2 = ConvPool(64, 128, 3, 2, 2, 2)
        self.convpool3 = ConvPool(128, 256, 3, 2, 2, 3)
        self.convpool4 = ConvPool(256, 512, 3, 2, 2, 3)
        self.convpool5 = ConvPool(512, 512, 3, 2, 2, 3)
        self.convpool5_shape = 512 * 7 * 7
        self.fc1 = paddle.nn.Linear(self.convpool5_shape, 4096)
        self.fc2 = paddle.nn.Linear(4096, 4096)
        self.fc3 = paddle.nn.Linear(4096, 3)

    def forward(self, inputs, label=None):

        x = self.convpool1(inputs)
        x = self.convpool2(x)
        x = self.convpool3(x)
        x = self.convpool4(x)
        x = self.convpool5(x)
        x = paddle.reshape(x, [-1, 512 * 7 * 7])
        x = self.fc1(x)
        x = self.fc2(x)
        out = self.fc3(x)

        if label is not None:
            acc = paddle.metric.accuracy(input=out, label=label)
            return out, acc
        else:
            return out

        # 实例化模型


model = VGGNet()
```

```
model.eval()
# 加载预训练模型参数
model.set_dict(paddle.load("work/checkpoints/save_dir_1500.pdparams"))

# 定义输入数据
input_spec = paddle.static.InputSpec(shape=[None, 3, 224, 224], dtype='float32', name='image')

# ONNX 模型导出
paddle.onnx.export(model, 'home', input_spec=[input_spec])

```

Train.py

```
import paddle
import matplotlib.pyplot as plt
from Dataset import dataset
from model import VGGNet
from data_processing import unzip_data, train_parameters, get_data_list

'''
参数初始化
'''
src_path = train_parameters['src_path']
target_path = train_parameters['target_path']
train_list_path = train_parameters['train_list_path']
eval_list_path = train_parameters['eval_list_path']

'''
解压原始数据到指定路径
'''
unzip_data(src_path, target_path)

'''
划分训练集与验证集，乱序，生成数据列表
'''
# 每次生成数据列表前，首先清空 train.txt 和 eval.txt
with open(train_list_path, 'w') as f:
    f.seek(0)
    f.truncate()
with open(eval_list_path, 'w') as f:
```

```
    f.seek(0)
    f.truncate()

# 生成数据列表
get_data_list(target_path, train_list_path, eval_list_path)

train_dataset = dataset('data', mode='train')
train_loader = paddle.io.DataLoader(train_dataset, batch_size=16,
    shuffle=True)

train_dataset.print_sample(200)
#print(train_dataset.__len__())


def draw_process(title, color, iters, data, label):
    plt.title(title, fontsize=24)
    plt.xlabel("iter", fontsize=20)
    plt.ylabel(label, fontsize=20)
    plt.plot(iters, data, color=color, label=label)
    plt.legend()
    plt.grid()
    plt.show()


model = VGGNet()
model.train()
cross_entropy = paddle.nn.CrossEntropyLoss()
optimizer = paddle.optimizer.Adam(learning_rate=train_parameters[
   'learning_strategy']['lr'],
                                  parameters=model.parameters())

steps = 0
Iters, total_loss, total_acc = [], [], []

for epo in range(train_parameters['num_epochs']):
    for _, data in enumerate(train_loader()):
        steps += 1
        x_data = data[0]
        y_data = data[1]
        predicts, acc = model(x_data, y_data)
        loss = cross_entropy(predicts, y_data)
        loss.backward()
        optimizer.step()
        optimizer.clear_grad()
```

```
        if steps % train_parameters["skip_steps"] == 0:
            Iters.append(steps)
            total_loss.append(loss.numpy()[0])
            total_acc.append(acc.numpy()[0])
            # 打印中间过程
            print('epo: {}, step: {}, loss is: {}, acc is: {}' \
                  .format(epo, steps, loss.numpy(), acc.numpy()))
        # 保存模型参数
        if steps % train_parameters["save_steps"] == 0:
            save_path = train_parameters["checkpoints"] + "/" + "save_dir_" + str(steps) + '.pdparams'
            print('save model to: ' + save_path)
            paddle.save(model.state_dict(), save_path)
paddle.save(model.state_dict(), train_parameters["checkpoints"] + "/" + "model_final.pdparams")
draw_process("trainning loss", "red", Iters, total_loss, "trainning loss")
draw_process("trainning acc", "green", Iters, total_acc, "trainning acc")

```

分割模型代码

TrainModel.ipynb 文件：

```
# 解压数据,数据来源：https://mosmed.ai/
#原本数据是 NiFit 格式数据，然后转换成 png 格式，输入 PaddleSeg 进行训练
!unzip data/data114821/MosMedSegPNG.zip -d /home/aistudio/work

#支持配置化训练和 API 方式训练,这里采用 API 方式进行训练
!pip install paddleseg SimpleITK

### 划分数据集
random.seed(1000)
path_origin = '/home/aistudio/work/MosMedSegPNG/origin'
files = list(filter(lambda x: x.endswith('.png'), os.listdir(path_origin)))
random.shuffle(files)
rate = int(len(files) * 0.8)#训练集和测试集 8：2
train_txt = open('/home/aistudio/work/MosMedSegPNG/train_list.txt','w')
```

```
val_txt = open('/home/aistudio/work/MosMedSegPNG/val_list.txt','w')
for i,f in enumerate(files):
    image_path = os.path.join(path_origin, f)
    label_path = image_path.replace("origin", "mask")
    if i < rate:
        train_txt.write(image_path + ' ' + label_path+ '\n')
    else:
        val_txt.write(image_path + ' ' + label_path+ '\n')
train_txt.close()
val_txt.close()

print('完成')
```

PyQt 代码

Deploy.py 部署文件

```
# -*- coding: utf-8 -*-
# pyuic5 -o SegGroundClassUI.py SegGroundClassUI.ui
import sys
from PyQt5.QtWidgets import *
from PyQt5.QtCore import *
from PyQt5.QtGui import *
from FormUI import Ui_Form

from paddleseg.models import BiSeNetV2
import paddleseg.transforms as T
from paddleseg.core import infer
import paddle
import numpy as np
import SimpleITK as sitk
import os
import cv2


def windowwc(sitkImage, ww=1500, wc=-550):
    """
    主要用于设置窗宽窗位
    @param sitkImage:SimpleITK 图像数据
    @param ww:窗宽窗位
    @param wc:窗宽窗位
    @return:sitkImage
    """
```

```
    min = int(wc - ww / 2.0)
    max = int(wc + ww / 2.0)
    intensityWindow = sitk.IntensityWindowingImageFilter()
    intensityWindow.SetWindowMaximum(max)
    intensityWindow.SetWindowMinimum(min)
    sitkImage = intensityWindow.Execute(sitkImage)
    return sitkImage


def readNii(path, ww, wc, isflipud=True, ):
    """
    读取和加载数据。如果图像是上下翻转的，就将其翻转过来
    @param path: 文件路径
    @param ww:窗宽窗位
    @param wc:窗宽窗位
    @param isflipud: 是否需要翻转
    @return: data
    """
    if type(path) == str:
        img = windowwc(sitk.ReadImage(path), ww, wc)
    else:
        img = windowwc(path, ww, wc)
    data = sitk.GetArrayFromImage(img)
    if isflipud:
        data = np.flip(data, 1)
    return data


class InferThread(QThread):
    """
    调用 PyQt5.QtCore，建立一个任务线程类，进行推理任务
    """
    # 收集推理失败的信号
    signal_infer_fail = pyqtSignal()
    # 传递推理结果
    signal_infer_result = pyqtSignal(np.ndarray)

    def __init__(self, sitkImage, model):
        super(InferThread, self).__init__()
        self.sitkImage = sitkImage
        self.model = model
        self.transforms = T.Compose([
            T.Resize(target_size=(512, 512)),
            T.Normalize()
```

```
        ])

    def run(self):
        """
            在启动线程后任务开始执行
        """
        try:
            data = readNii(self.sitkImage, 1500, -500)
            inferData = np.zeros_like(data)
            d, h, w = data.shape

            for i in range(d):
                img = data[i].copy()
                img = img.astype(np.float32)
                pre = self.nn_infer(self.model, img, self.transfo
rms)
                inferData[i] = pre

            self.signal_infer_result.emit(inferData)
        except Exception as e:
            print(e)
            self.signal_infer_fail.emit()

    def nn_infer(self, model, im, transforms):
        """
        预测结果
        @param model: 模型参数
        @param im: 图像数据
        @param transforms:传入 transforms 方法
        @return: 预测结果 pred
         """
         img, _ = transforms(im)
         img = paddle.to_tensor(img[np.newaxis, :])
         pre = infer.inference(model, img)
         pred = paddle.argmax(pre, axis=1).numpy().reshape((512,
512))
         return pred.astype('uint8')


class MainWindow(QWidget, Ui_Form):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.setupUi(self)

```

```
113.         self.initUI()
114.         self.setWindowTitle('肺炎检测助手')
115.         # 打开 nii 文件选择器
116.         self.bn_open.clicked.connect(self.openFile)
117.         # 打开模型文件选择器
118.         self.bn_loadModel.clicked.connect(self.openModleFile)
119.         # 推理按钮
120.         self.bn_infer.clicked.connect(self.infer)
121.         self.bn_output.clicked.connect(self.outputFile)
122.
123.         self.sitkImage = object()
124.         self.npImage = object()
125.         # 记录当前第几层
126.         self.currIndex = 0
127.         # 记录数据的最大层
128.         self.maxCurrIndex = 0
129.         # 记录数据的最小层，其实就是 0
130.         self.minCurrIndex = 0
131.         self.baseFileName = ''
132.         self.isInferSucceed = False
133.
134.         # 判断是否按下鼠标右键
135.         self.isRightPressed = bool(False)
136.
137.         self.model = object()
138.
139.         # 判断模型是否加载成功
140.         self.isModelReady = False
141.
142.         # 宽宽窗位滑动条
143.         self.slider_ww.valueChanged.connect(self.resetWWWcAndSho
    w)
144.         self.slider_wc.valueChanged.connect(self.resetWWWcAndSho
    w)
145.
146.         # 设置窗宽窗位文本框只能输入一定范围的整数
147.         intValidator = QIntValidator(self)
148.         intValidator.setRange(-2000, 2000)
149.         self.line_ww.setValidator(intValidator)
150.         self.line_ww.editingFinished.connect(self.resetWWWcAndSh
    ow)
151.         self.line_wc.setValidator(intValidator)
152.         self.line_wc.editingFinished.connect(self.resetWWWcAndSh
    ow)
```

```
153.
154.         self.listWidget.itemDoubleClicked.connect(self.changeLay
    er)
155.
156.     def initUI(self):
157.         try:
158.             # 定义展示的窗体及其初始的参数
159.             self.wwwcList = {'肺窗': [1700, -700]}
160.
161.             self.line_ww.setText(str(1700))
162.             self.line_wc.setText(str(-700))
163.
164.             self.slider_ww.setValue(1700)
165.             self.slider_wc.setValue(-700)
166.             self.ww = 1700
167.             self.wc = -700
168.
169.             self.currWw = self.ww
170.             self.currWc = self.wc
171.         except Exception as e:
172.             print(e)
173.
174.     def openFile(self):
175.         """
176.         打开医学影像文件选择器
177.         """
178.         try:
179.             filename, _ = QFileDialog.getOpenFileName(self,
180.                                                       "选取文件",
181.                                                       "./",
182.                                                       "Nii Files
    (*.nii);;Nii Files (*.nii.gz);;All Files (*)")
183.             if filename:
184.                 # 清空列表
185.                 self.listWidget.clear()
186.                 self.isInferSucceed = False
187.                 self.text_loadModel.setText("数据加载完毕")
188.                 self.baseFileName = os.path.basename(filename).s
    plit('.')[0]
189.                 self.sitkImage = sitk.ReadImage(filename)
190.                 self.npImage = readNii(self.sitkImage, self.ww,
    self.wc)
191.                 self.maxCurrIndex = self.npImage.shape[0]
192.                 self.currIndex = int(self.maxCurrIndex / 2)
```

```
193.                 self.showImg(self.npImage[self.currIndex])
194.         except Exception as e:
195.             print(e)
196.
197.     def openModleFile(self):
198.         """
199.         打开模型文件选择器
200.         """
201.         filename, _ = QFileDialog.getOpenFileName(self, "选取文件
    ", "./", "model Files (*.pdparams)")
202.
203.         if filename:
204.             try:
205.                 self.text_loadModel.setText(" ")
206.                 num_class = int(2)
207.                 self.model = BiSeNetV2(num_classes=num_class)
208.                 para_state_dict = paddle.load(filename)
209.                 self.model.set_dict(para_state_dict)
210.                 self.text_loadModel.setText("模型加载完毕")
211.                 self.isModelReady = True
212.             except Exception as e:
213.                 self.text_loadModel.setText("模型加载失败")
214.                 print(e)
215.
216.     def wheelEvent(self, event):
217.         """
218.         定义鼠标滑轮事件
219.         """
220.         try:
221.             if self.maxCurrIndex != self.minCurrIndex:
222.                 self.angle = event.angleDelta() / 8
223.                 self.angleY = self.angle.y()
224.                 if self.angleY > 0:
225.                     if self.currIndex < self.maxCurrIndex - 1:
226.                         self.currIndex += 1
227.                         if self.isInferSucceed:
228.                             self.showImg(self.drawContours(self.
    npImage, self.inferData, self.currIndex))
229.                         else:
230.                             self.showImg(self.npImage[self.currI
    ndex])
231.                 elif self.angleY < 0:
232.                     if self.currIndex != self.minCurrIndex:
233.                         self.currIndex -= 1
```

```
234.                         if self.isInferSucceed:
235.                             # self.npImage = self.drawContours(s
    elf.npImage, self.inferData)
236.                             self.showImg(self.drawContours(self.
    npImage, self.inferData, self.currIndex))
237.                         else:
238.                             self.showImg(self.npImage[self.currI
    ndex])
239.         except Exception as e:
240.             print(e)
241.
242.     def mousePressEvent(self, event):
243.         """
244.             重载鼠标单机事件
245.         """
246.         # 左键按下
247.         if event.buttons() == Qt.RightButton:
248.             # 左键按下(图片被点住),置 Ture
249.             self.isRightPressed = True
250.             self.preMousePosition = event.pos()
251.         elif event.buttons() == Qt.MidButton | Qt.RightButton:
252.             self.isRightPressed = False
253.
254.     def mouseReleaseEvent(self, event):
255.         if event.button() == Qt.RightButton:
256.             self.isRightPressed = False
257.
258.     def mouseMoveEvent(self, event):
259.         """
260.         重载一下鼠标移动事件
261.         """
262.         try:
263.             if self.maxCurrIndex != self.minCurrIndex:
264.                 # 右键按下
265.                 if self.isRightPressed:
266.                     # 鼠标当前位置-先前位置=单次偏移量
267.                     self.endMousePosition = event.pos() - self.p
    reMousePosition
268.                     self.preMousePosition = event.pos()
269.                     ww = self.endMousePosition.x() + self.currWw
270.                     wc = self.endMousePosition.y() + self.currWc
271.                     if ww < -2000:
272.                         ww = -2000
273.                     elif ww > 2000:
```

```
                        ww = 2000
                    if wc < -2000:
                        wc = -2000
                    elif wc > 2000:
                        wc = 2000
                    self.currWw = ww
                    self.currWc = wc
                    self.slider_ww.setValue(int(self.currWw))
                    self.slider_wc.setValue(int(self.currWc))
                    self.line_ww.setText(str(self.currWw))
                    self.line_wc.setText(str(self.currWc))
                    self.resetWWWcAndShow()
            except Exception as e:
                print(e)

    def showImg(self, img):
        """
        显示图片
        @param img: 待显示的图片
        """
        try:
            if img.ndim == 2:
                img = np.expand_dims(img, axis=2)
                img = np.concatenate((img, img, img), axis=-1).astype(np.uint8)
            elif img.ndim == 3:
                img = img.astype(np.uint8)
            qimage = QImage(img, img.shape[0], img.shape[1], img.shape[1] * 3, QImage.Format_RGB888)
            pixmap_imgSrc = QPixmap.fromImage(qimage)

            self.canvas.setPixmap(pixmap_imgSrc)
        except Exception as e:
            print(e)

    def resetWWWcAndShow(self):
        """
        通过四个方式可以修改医学图像的窗宽窗位，每次修改后都会在界面呈现出来
        """
        if hasattr(self.sender(), "objectName"):
            objectName = self.sender().objectName()
        else:
            objectName = None
```

```
315.         try:
316.             if objectName == '':
317.                 self.line_ww.setText(str(1700))
318.                 self.line_wc.setText(str(-700))
319.                 self.slider_ww.setValue(1700)
320.                 self.slider_wc.setValue(-700)
321.                 self.ww = 1700
322.                 self.wc = -700
323.                 self.currWw = self.ww
324.                 self.currWc = self.wc
325.             if objectName == 'slider_ww' or objectName == 'slide
    r_wc':
326.                 self.currWw = self.slider_ww.value()
327.                 self.currWc = self.slider_wc.value()
328.                 self.line_ww.setText(str(self.currWw))
329.                 self.line_wc.setText(str(self.currWc))
330.             elif objectName == 'line_ww' or objectName == 'line_
    wc':
331.                 self.currWw = int(self.line_ww.text())
332.                 self.currWc = int(self.line_wc.text())
333.                 self.slider_ww.setValue(self.currWw)
334.                 self.slider_wc.setValue(self.currWc)
335.             if self.maxCurrIndex != self.minCurrIndex:
336.                 self.npImage = readNii(self.sitkImage, self.curr
    Ww, self.currWc)
337.                 if self.isInferSucceed:
338.                     self.showImg(self.drawContours(self.npImage,
     self.inferData, self.currIndex))
339.                 else:
340.                     self.showImg(self.npImage[self.currIndex])
341.         except Exception as e:
342.             print(e)
343.
344.     def infer(self):
345.         """
346.             模型分割预测
347.         """
348.         if self.maxCurrIndex != self.minCurrIndex and self.isMod
    elReady:
349.             self.bn_infer.setEnabled(True)
350.             # 创建推理线程
351.             self.infer_thread = InferThread(self.sitkImage, self.
    model)
352.             # 绑定推理失败的槽函数
```

```
353.             self.infer_thread.signal_infer_fail.connect(self.inf
    er_fail)
354.             # 绑定推理成功的槽函数
355.             self.infer_thread.signal_infer_result.connect(self.i
    nfer_result)
356.             self.infer_thread.start()
357.             self.text_loadModel.setText("正在推理中·····")
358.
359.         else:
360.             QMessageBox.warning(self, "提示", "推理失败，推理前请先
    确保:\n1.加载模型\n2.加载数据", QMessageBox.Yes, QMessageBox.Yes)
361.
362.     def infer_result(self, inferData):
363.         """
364.             分割模型预测成功后，结果保存在 self.inferData
365.             @param inferData: 推理数据
366.         """
367.         # 推理成功，并显示结果
368.         try:
369.             self.inferData = inferData.astype(np.uint8)
370.             QMessageBox.information(self, "提示", "模型推理成
    功!", QMessageBox.Yes, QMessageBox.Yes)
371.             self.text_loadModel.setText("推理完毕")
372.             self.isInferSucceed = True
373.             self.infer_thread.quit()
374.             self.addListInfo(self.inferData)
375.             self.showImg(self.drawContours(self.npImage, self.in
    ferData, self.currIndex))
376.         except Exception as e:
377.             print(e)
378.
379.     def infer_fail(self):
380.         """
381.             如果推理失败，则报错
382.         """
383.         QMessageBox.warning(self, "警告", "模型推理失
    败!", QMessageBox.Yes, QMessageBox.Yes)
384.
385.     def outputFile(self):
386.         """
387.             将保存模型预测结果为 nii 格式文件
388.         """
389.         try:
390.             if self.isInferSucceed:
```

```
391.                 filedir = QFileDialog.getExistingDirectory(None, "文件保存", os.getcwd())
392.                 if filedir:
393.                     # 读取 nii 文件时转换 np 文件时对数据进行上下翻转，再输入模型推理，保存回 nii 文件时要翻转回来。
394.                     self.inferData = np.flip(self.inferData, 1)
395.                     pre_sitkImage = sitk.GetImageFromArray(self.inferData)
396.                     pre_sitkImage.CopyInformation(self.sitkImage)
397.                     pre_sitkImage = sitk.Cast(pre_sitkImage, sitk.sitkUInt8)
398.                     save_path = os.path.join(filedir, self.baseFileName + '_mask.nii')
399.                     sitk.WriteImage(pre_sitkImage, save_path)
400.             else:
401.                 QMessageBox.warning(self, "警告", "无进行过推理，无法保存！", QMessageBox.Yes, QMessageBox.Yes)
402.         except Exception as e:
403.             print(e)
404.
405.     def drawContours(self, npImage, inferData, currIndex):
406.         """
407.             通过 OpenCV 将 mask 转换成轮廓绘制在原图上
408.             @param npImage: 图像数据
409.             @param inferData: 模型推理的结果
410.             @param currIndex: 层数序号
411.             @return: 绘制轮廓后的图片
412.         """
413.         img = npImage[currIndex]
414.         img = np.expand_dims(img, axis=2)
415.         img = np.concatenate((img, img, img), axis=-1).astype(np.uint8)
416.         ret, thresh = cv2.threshold(inferData[currIndex], 0, 255, cv2.THRESH_BINARY)
417.         thresh = cv2.dilate(thresh, kernel=np.ones((5, 5), np.uint8), iterations=1)
418.         contours, hierarchy = cv2.findContours(thresh, 1, 2)
419.         # 绘制轮廓过程
420.         img = cv2.drawContours(img, contours, -1, (0, 255, 0), 1)
421.
422.         return img
423.
```

```
424.     def addListInfo(self, inferData):
425.         """
426.             增加列表信息
427.             @param inferData:模型推理的结果
428.         """
429.         self.listWidget.clear()
430.         d, h, w = inferData.shape
431.         result = {}
432.         for i in range(d):
433.             img = inferData[i]
434.             if np.sum(img > 0) != 0:
435.                 result[str(i)] = np.sum(img > 0)
436.
437.         result = sorted(result.items(), key=lambda x: x[1], reve
    rse=True)
438.         for key, value in result:
439.             self.listWidget.addItem("层 " + str(int(key) + 1))
440.
441.     def changeLayer(self, item):
442.         """
443.         点击列表自动展示该层
444.         @param item: 控制层数的对象
445.         """
446.         self.currIndex = int(item.text().split(' ')[1]) - 1
447.         if self.isInferSucceed:
448.             self.showImg(self.drawContours(self.npImage, self.in
    ferData, self.currIndex))
449.         else:
450.             self.showImg(self.npImage[self.currIndex])
451.
452.
453. if __name__ == "__main__":
454.     app = QApplication(sys.argv)
455.     main = MainWindow()
456.     main.show()
457.     sys.exit(app.exec_())
458.
```

FormUI.py 文件

```
1. # -*- coding: utf-8 -*-
2.
```

```
3. # Form implementation generated from reading ui file 'SegGroundCl
   assUI1.ui'
4. #
5. # Created by: PyQt5 UI code generator 5.15.4
6. #
7. # WARNING: Any manual changes made to this file will be lost when
    pyuic5 is
8. # run again.  Do not edit this file unless you know what you are
   doing.
9.
10.
11.from PyQt5 import QtCore, QtGui, QtWidgets
12.
13.
14.class Ui_Form(object):
15.    def setupUi(self, Form):
16.        """
17.        定义窗体对象，并且设置基本参数
18.        @param Form: 窗体对象
19.        """
20.        Form.setObjectName("Form")
21.        Form.resize(1286, 794)
22.        Form.setMaximumSize(QtCore.QSize(1286, 794))
23.        self.gridLayout_4 = QtWidgets.QGridLayout(Form)
24.        self.gridLayout_4.setObjectName("gridLayout_4")
25.        self.gridLayout_5 = QtWidgets.QGridLayout()
26.        self.gridLayout_5.setObjectName("gridLayout_5")
27.        self.canvas = QtWidgets.QLabel(Form)
28.        self.canvas.setMinimumSize(QtCore.QSize(512, 512))
29.        self.canvas.setMaximumSize(QtCore.QSize(16777215, 1677721
   5))
30.        font = QtGui.QFont()
31.        font.setPointSize(10)
32.        self.canvas.setFont(font)
33.        self.canvas.setMouseTracking(False)
34.        self.canvas.setAutoFillBackground(False)
35.        self.canvas.setFrameShadow(QtWidgets.QFrame.Plain)
36.        self.canvas.setLineWidth(1)
37.        self.canvas.setMidLineWidth(0)
38.        self.canvas.setText("")
39.        self.canvas.setObjectName("canvas")
40.        self.gridLayout_5.addWidget(self.canvas, 0, 1, 1, 1)
41.        self.gridLayout_8 = QtWidgets.QGridLayout()
42.        self.gridLayout_8.setObjectName("gridLayout_8")
```

```
43.         self.gridLayout_6 = QtWidgets.QGridLayout()
44.         self.gridLayout_6.setObjectName("gridLayout_6")
45.         self.groupBox_2 = QtWidgets.QGroupBox(Form)
46.         self.groupBox_2.setMinimumSize(QtCore.QSize(0, 0))
47.         self.groupBox_2.setMaximumSize(QtCore.QSize(16777215, 167
    77215))
48.         font = QtGui.QFont()
49.         font.setPointSize(10)
50.         self.groupBox_2.setFont(font)
51.         self.groupBox_2.setAlignment(QtCore.Qt.AlignCenter)
52.         self.groupBox_2.setFlat(True)
53.         self.groupBox_2.setObjectName("groupBox_2")
54.         self.gridLayout_9 = QtWidgets.QGridLayout(self.groupBox_2
    )
55.         self.gridLayout_9.setObjectName("gridLayout_9")
56.         self.verticalLayout_2 = QtWidgets.QVBoxLayout()
57.         self.verticalLayout_2.setObjectName("verticalLayout_2")
58.         self.horizontalLayout = QtWidgets.QHBoxLayout()
59.         self.horizontalLayout.setObjectName("horizontalLayout")
60.         self.label = QtWidgets.QLabel(self.groupBox_2)
61.         self.label.setObjectName("label")
62.         self.horizontalLayout.addWidget(self.label)
63.         self.line_ww = QtWidgets.QLineEdit(self.groupBox_2)
64.         self.line_ww.setMaximumSize(QtCore.QSize(100, 16777215))
65.         self.line_ww.setObjectName("line_ww")
66.         self.horizontalLayout.addWidget(self.line_ww)
67.         self.verticalLayout_2.addLayout(self.horizontalLayout)
68.         self.slider_ww = QtWidgets.QSlider(self.groupBox_2)
69.         self.slider_ww.setMinimum(-2000)
70.         self.slider_ww.setMaximum(2000)
71.         self.slider_ww.setOrientation(QtCore.Qt.Horizontal)
72.         self.slider_ww.setObjectName("slider_ww")
73.         self.verticalLayout_2.addWidget(self.slider_ww)
74.         self.horizontalLayout_2 = QtWidgets.QHBoxLayout()
75.         self.horizontalLayout_2.setObjectName("horizontalLayout_2
    ")
76.         self.label_5 = QtWidgets.QLabel(self.groupBox_2)
77.         self.label_5.setObjectName("label_5")
78.         self.horizontalLayout_2.addWidget(self.label_5)
79.         self.line_wc = QtWidgets.QLineEdit(self.groupBox_2)
80.         self.line_wc.setMaximumSize(QtCore.QSize(100, 16777215))
81.         self.line_wc.setObjectName("line_wc")
82.         self.horizontalLayout_2.addWidget(self.line_wc)
83.         self.verticalLayout_2.addLayout(self.horizontalLayout_2)
```

```
84.         self.slider_wc = QtWidgets.QSlider(self.groupBox_2)
85.         self.slider_wc.setMinimum(-2000)
86.         self.slider_wc.setMaximum(2000)
87.         self.slider_wc.setOrientation(QtCore.Qt.Horizontal)
88.         self.slider_wc.setObjectName("slider_wc")
89.         self.verticalLayout_2.addWidget(self.slider_wc)
90.         self.gridLayout_9.addLayout(self.verticalLayout_2, 0, 0,
    1, 1)
91.         self.gridLayout_6.addWidget(self.groupBox_2, 2, 0, 1, 1)
92.         self.groupBox = QtWidgets.QGroupBox(Form)
93.         self.groupBox.setEnabled(True)
94.         self.groupBox.setMinimumSize(QtCore.QSize(250, 0))
95.         self.groupBox.setMaximumSize(QtCore.QSize(16777215, 16777
    215))
96.         font = QtGui.QFont()
97.         font.setPointSize(10)
98.         self.groupBox.setFont(font)
99.         self.groupBox.setLayoutDirection(QtCore.Qt.LeftToRight)
100.         self.groupBox.setInputMethodHints(QtCore.Qt.ImhNone)
101.         self.groupBox.setAlignment(QtCore.Qt.AlignCenter)
102.         self.groupBox.setFlat(True)
103.         self.groupBox.setCheckable(False)
104.         self.groupBox.setObjectName("groupBox")
105.         self.gridLayout = QtWidgets.QGridLayout(self.groupBox)
106.         self.gridLayout.setObjectName("gridLayout")
107.         self.bn_open = QtWidgets.QPushButton(self.groupBox)
108.         self.bn_open.setObjectName("bn_open")
109.         self.gridLayout.addWidget(self.bn_open, 0, 0, 1, 1)
110.         self.bn_output = QtWidgets.QPushButton(self.groupBox)
111.         self.bn_output.setMaximumSize(QtCore.QSize(16777215, 167
    77215))
112.         self.bn_output.setObjectName("bn_output")
113.         self.gridLayout.addWidget(self.bn_output, 4, 0, 1, 1)
114.         self.bn_infer = QtWidgets.QPushButton(self.groupBox)
115.         self.bn_infer.setObjectName("bn_infer")
116.         self.gridLayout.addWidget(self.bn_infer, 2, 0, 1, 1)
117.         self.bn_loadModel = QtWidgets.QPushButton(self.groupBox)
118.         self.bn_loadModel.setMaximumSize(QtCore.QSize(16777215,
    16777215))
119.         self.bn_loadModel.setObjectName("bn_loadModel")
120.         self.gridLayout.addWidget(self.bn_loadModel, 1, 0, 1, 1)
121.         self.text_loadModel = QtWidgets.QLabel(self.groupBox)
122.         self.text_loadModel.setText("")
123.         self.text_loadModel.setObjectName("text_loadModel")
```

```
124.         self.gridLayout.addWidget(self.text_loadModel, 5, 0, 1,
   1)
125.         self.gridLayout_6.addWidget(self.groupBox, 0, 0, 1, 1)
126.         self.groupBox_3 = QtWidgets.QGroupBox(Form)
127.         self.groupBox_3.setMinimumSize(QtCore.QSize(0, 0))
128.         self.groupBox_3.setMaximumSize(QtCore.QSize(16777215, 16
   777215))
129.         font = QtGui.QFont()
130.         font.setPointSize(10)
131.         self.groupBox_3.setFont(font)
132.         self.groupBox_3.setAlignment(QtCore.Qt.AlignCenter)
133.         self.groupBox_3.setObjectName("groupBox_3")
134.         self.gridLayout_2 = QtWidgets.QGridLayout(self.groupBox_
   3)
135.         self.gridLayout_2.setObjectName("gridLayout_2")
136.         self.listWidget = QtWidgets.QListWidget(self.groupBox_3)
137.         self.listWidget.setObjectName("listWidget")
138.         self.gridLayout_2.addWidget(self.listWidget, 0, 0, 1, 1)
139.         self.gridLayout_6.addWidget(self.groupBox_3, 3, 0, 1, 1)
140.         self.gridLayout_8.addLayout(self.gridLayout_6, 0, 2, 1,
   1)
141.         self.gridLayout_5.addLayout(self.gridLayout_8, 0, 3, 1,
   1)
142.         spacerItem = QtWidgets.QSpacerItem(225, 20, QtWidgets.QS
   izePolicy.Fixed, QtWidgets.QSizePolicy.Minimum)
143.         self.gridLayout_5.addItem(spacerItem, 0, 2, 1, 1)
144.         self.gridLayout_4.addLayout(self.gridLayout_5, 0, 1, 1,
   1)
145.         spacerItem1 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QS
   izePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
146.         self.gridLayout_4.addItem(spacerItem1, 0, 0, 1, 1)
147.
148.         self.retranslateUi(Form)
149.         QtCore.QMetaObject.connectSlotsByName(Form)
150.
151.     def retranslateUi(self, Form):
152.         """
153.         定义显示的文字
154.         @param Form: 窗体对象
155.         """
156.         _translate = QtCore.QCoreApplication.translate
157.         Form.setWindowTitle(_translate("Form", "Form"))
158.         self.groupBox_2.setTitle(_translate("Form", "肺窗"))
159.         self.label.setText(_translate("Form", "窗宽："))
```

```
        self.label_5.setText(_translate("Form", "窗位："))
        self.groupBox.setTitle(_translate("Form", "模型:BiSeNetV2"))
        self.bn_open.setText(_translate("Form", "加载数据"))
        self.bn_output.setText(_translate("Form", "导出 Mask"))
        self.bn_infer.setText(_translate("Form", "推理"))
        self.bn_loadModel.setText(_translate("Form", "加载模型"))
        self.groupBox_3.setTitle(_translate("Form", "位置"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()
    ui.setupUi(Form)
    Form.show()
    sys.exit(app.exec_())

```

C#部署代码

PAI.cs 文件

```
using System;
using System.Windows.Forms;

namespace WindowsFormsApp1
{
    public partial class PAI : Form
    {
        //public static Form1 form = new Form1();
        Form1 _form1;


        public PAI(Form1 form1)
        {
            InitializeComponent();

            _form1 = form1;
        }

```

```
        private void button1_Click(object sender, EventArgs e)
        {
            //form.Show();

        }

        private void linkLabel1_LinkClicked(object sender, LinkLa
   belLinkClickedEventArgs e)
        {
            linkLabel1.LinkVisited = true;
            _form1.Show();
            this.Hide();
        }

        private void PAI_Load(object sender, EventArgs e)
        {

        }
    }
}
```

Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Microsoft.ML.OnnxRuntime.Tensors;  // DenseTensor
using SixLabors.ImageSharp;  // Image, Size
using SixLabors.ImageSharp.PixelFormats;  // Rgb24
using SixLabors.ImageSharp.Processing;  // image.Mutate

namespace WindowsFormsApp1
{
    public partial class Form1 : Form
    {
        public static Xray XrayForm;
```

```
21.         public static PAI PAIForm;
22.
23.         public Form1()
24.         {
25.             InitializeComponent();
26.             XrayForm = new Xray(this);
27.             PAIForm = new PAI(this);
28.
29.         }
30.
31.         private void Form1_Load(object sender, EventArgs e)
32.         {
33.
34.         }
35.
36.         private void toolStripTextBox1_Click(object sender, Event
    Args e)
37.         {
38.
39.         }
40.
41.         private void toolStripLabel1_Click(object sender, EventAr
    gs e)
42.         {
43.
44.         }
45.
46.         private void toolStripComboBox1_Click(object sender, Even
    tArgs e)
47.         {
48.
49.         }
50.
51.         private void toolStripProgressBar1_Click(object sender, E
    ventArgs e)
52.         {
53.
54.         }
55.
56.         private void toolStrip1_ItemClicked(object sender, ToolSt
    ripItemClickedEventArgs e)
57.         {
58.
59.         }
```

```
60.
61.         private void CT_Click(object sender, EventArgs e)
62.         {
63.
64.         }
65.
66.         private void label1_Click(object sender, EventArgs e)
67.         {
68.
69.
70.         }
71.
72.         private void label4_Click(object sender, EventArgs e)
73.         {
74.
75.         }
76.
77.         private void linkLabel4_LinkClicked(object sender, LinkLa
    belLinkClickedEventArgs e)
78.         {
79.             linkLabel4.LinkVisited = true;
80.             XrayForm.Show();
81.             this.Hide();
82.         }
83.
84.         private void linkLabel2_LinkClicked(object sender, LinkLa
    belLinkClickedEventArgs e)
85.         {
86.             linkLabel2.LinkVisited = true;
87.             PAIForm.Show();
88.             this.Hide();
89.
90.         }
91.
92.         private void linkLabel1_LinkClicked(object sender, LinkLa
    belLinkClickedEventArgs e)
93.         {
94.             this.linkLabel1.Text = "https://github.com/Laymanpyth
    on/OpenCV_Competition";
95.             System.Diagnostics.Process.Start(this.linkLabel1.Text)
    ;//打开网页
96.
97.         }
98.
```

```
99.         private void linkLabel5_LinkClicked(object sender, LinkLa
    belLinkClickedEventArgs e)
100.         {
101.             linkLabel5.LinkVisited = true;
102.             Process process = new Process();
103.             process.StartInfo.FileName = "cmd.exe";
104.             process.StartInfo.Arguments = "/c" + "python D:\\Ass
    istant\\pyqt5\\Deploy.py";
105.             process.StartInfo.UseShellExecute = false;   //是否使
    用操作系统shell启动
106.             process.StartInfo.CreateNoWindow = true;   //是否在新
    窗口中启动该进程的值（不显示程序窗口）
107.             process.Start();
108.             process.WaitForExit();  //等待程序执行完退出进程
109.             process.Close();
110.
111.             //this.Hide();
112.
113.         }
114.
115.         private void label2_Click(object sender, EventArgs e)
116.         {
117.
118.         }
119.
120.         private void label3_Click(object sender, EventArgs e)
121.         {
122.
123.         }
124.     }
125. }
```

Xray.cs

```
1. using System;
2. using System.Collections.Generic;
3. using System.Data;
4. using System.Linq;
5. using System.Windows.Forms;
6. using System.IO;
7. using Microsoft.ML.OnnxRuntime.Tensors;
8. using SixLabors.ImageSharp;
9. using SixLabors.ImageSharp.Processing;
```

```
10. using SixLabors.ImageSharp.PixelFormats;
11. using Microsoft.ML.OnnxRuntime;
12. 
13. 
14. namespace WindowsFormsApp1
15. {
16.     public partial class Xray : Form
17.     {
18.         Form1 _form1;
19. 
20. 
21.         public Xray(Form1 form1)
22.         {
23.             InitializeComponent();
24. 
25.             _form1 = form1;
26.         }
27. 
28.         private void checkBox1_CheckedChanged(object sender, Even
    tArgs e)
29.         {
30. 
31.         }
32. 
33.         private void Xray_Load(object sender, EventArgs e)
34.         {
35. 
36.         }
37. 
38.         private void label5_Click(object sender, EventArgs e)
39.         {
40. 
41.         }
42. 
43.         private void button2_Click(object sender, EventArgs e)
44.         {
45.             separation sep = new separation();
46.             string score = processing_score();
47.             string Tag = processing();
48. 
49.             textBox1.Text = score;
50.            // textBox4.Text = gender;
51.            // textBox3.Text = IDnumber;
52.             textBox3.Text = Tag;
```

```
53.         }
54.
55.         private void button1_Click(object sender, EventArgs e)
56.         {
57.             System.Threading.Thread s = new System.Threading.Thre
    ad(new System.Threading.ThreadStart(Getpicture));
58.
59.             s.ApartmentState = System.Threading.ApartmentState.ST
    A;
60.             s.Start();
61.
62.         }
63.
64.
65.         private void Getpicture()
66.         {
67.             //string path = System.AppDomain.CurrentDomain.BaseDi
    rectory;
68.             OpenFileDialog ofd = new OpenFileDialog();
69.             //ofd.InitialDirectory = @"D:/";
70.             ofd.Filter = "jpeg 文件(*.jpg)|*.jpg|png 文件
    (*.png)|*.png";
71.             ofd.FilterIndex = 1;
72.             ofd.RestoreDirectory = true;
73.             ofd.Title = "选择 X 光胸片";
74.             string path;
75.             if (ofd.ShowDialog() == DialogResult.OK)
76.             {
77.                 path = ofd.FileName;
78.                 string pathname = System.AppDomain.CurrentDomain.
    BaseDirectory;
79.                 using (Stream stream = ofd.OpenFile())
80.                 {
81.                     using (FileStream fs = new FileStream(pathnam
    e + "张三
    _114514_M.jpg", FileMode.OpenOrCreate, FileAccess.ReadWrite))
82.                     {
83.                         stream.CopyTo(fs);
84.                         fs.Flush();
85.                     }
86.                 }
87.                 this.pictureBox1.SizeMode = PictureBoxSizeMode.St
    retchImage;
```

```
88.                 pictureBox1.Image = System.Drawing.Image.FromFile
    (path);
89.             }
90.             string imageFilePath = @"张三_114514_M.jpg";
91.             // Read image
92.             // Rgb24:
    Pixel type containing three 8-bit unsigned normalized values rang
    ing from 0 to
93.             //        255. The color components are stored in red,
     green, blue order
94.             // SixLabors.ImageSharp.Image
95.         }
96.
97.         private string processing()
98.         {
99.             string modelFilePath = @"D:\Assistant\WindowsFormsApp
    1\home.onnx";
100.             separation sep = new separation();
101.             string filename = sep.Separate();
102.             string imageFilePath = System.AppDomain.CurrentDomai
    n.BaseDirectory + filename;
103.             using Image<Rgb24> image = SixLabors.ImageSharp.Imag
    e.Load<Rgb24>(imageFilePath);  // 以rgb 形式读取图片
104.
                                    // Resize image
105.
                                    // Resize image
106.             image.Mutate(x =>
107.             {
108.                 x.Resize(new ResizeOptions
109.                 {
110.                     Size = new SixLabors.ImageSharp.Size(224, 22
    4),
111.                     Mode = ResizeMode.Crop
112.                 });
113.             });
114.
115.             image.Mutate(x =>
116.                     x.Resize(224, 224)
117.             );
118.
119.
120.             // Preprocess image
121.
```

```
            Tensor<float> input = new DenseTensor<float>(new[] { 1, 3, 224, 224 });
            var mean = new[] { 0.485f, 0.456f, 0.406f };
            var stddev = new[] { 0.229f, 0.224f, 0.225f };
            for (int y = 0; y < 224; y++)
            {
                image.ProcessPixelRows(im =>
                {
                    var pixelSpan = im.GetRowSpan(y);
                    for (int x = 0; x < 224; x++)
                    {
                        input[0, 0, y, x] = pixelSpan[x].R / 255f;
                        input[0, 1, y, x] = pixelSpan[x].G / 255f;
                        input[0, 2, y, x] = pixelSpan[x].B / 255f;
                    }
                });

            }


            // Setup inputs
            var inputs = new List<NamedOnnxValue>
            {
               NamedOnnxValue.CreateFromTensor("image", input)
            };

            // Run inference
            using var session = new InferenceSession(modelFilePath);
            using IDisposableReadOnlyCollection<DisposableNamedOnnxValue> results = session.Run(inputs);

            // Postprocess to get softmax vector
            IEnumerable<float> output = results.First().AsEnumerable<float>();  // First(): The first element in the specified sequence. AsEnumerable:
            float sum = output.Sum(x => (float)Math.Exp(x));   // sum(e^x)
            IEnumerable<float> softmax = output.Select(x => (float)Math.Exp(x) / sum);  // e^x / sum
```

```
156.
157.             //Extract top 1 predicted classes
158.             IEnumerable<Prediction> top1 = softmax.Select((x, i)
    => new Prediction { Label = LabelMap.Labels[i], Confidence = x }
   )
159.             .OrderByDescending(x => x.Confidence)
160.             .Take(1);
161.             string Tag = "none";
162.             foreach (var t in top1)
163.             {
164.                 string c = t.Label;
165.                 Tag = c;
166.             }
167.
168.             return Tag;
169.         }
170.
171.         private string processing_score()
172.         {
173.             string modelFilePath = @"D:\Assistant\WindowsFormsAp
   p1\home.onnx";
174.             separation sep = new separation();
175.             string filename = sep.Separate();
176.             string imageFilePath = System.AppDomain.CurrentDomai
   n.BaseDirectory + filename;
177.             using Image<Rgb24> image = SixLabors.ImageSharp.Imag
   e.Load<Rgb24>(imageFilePath);  // 以rgb形式读取图片
178.
                                    // Resize image
179.
                                    // Resize image
180.             image.Mutate(x =>
181.             {
182.                 x.Resize(new ResizeOptions
183.                 {
184.                     Size = new SixLabors.ImageSharp.Size(224, 22
   4),
185.                     Mode = ResizeMode.Crop
186.                 });
187.             });
188.
189.             image.Mutate(x =>
190.                     x.Resize(224, 224)
191.             );
```

```
192.
193.
194.             // Preprocess image
195.
196.             Tensor<float> input = new DenseTensor<float>(new[] {
    1, 3, 224, 224 });
197.             var mean = new[] { 0.485f, 0.456f, 0.406f };
198.             var stddev = new[] { 0.229f, 0.224f, 0.225f };
199.             for (int y = 0; y < 224; y++)
200.             {
201.                 image.ProcessPixelRows(im =>
202.                 {
203.                     var pixelSpan = im.GetRowSpan(y);
204.                     for (int x = 0; x < 224; x++)
205.                     {
206.                         input[0, 0, y, x] = pixelSpan[x].R / 255
    f;
207.                         input[0, 1, y, x] = pixelSpan[x].G / 255
    f;
208.                         input[0, 2, y, x] = pixelSpan[x].B / 255
    f;
209.                     }
210.                 });
211.
212.             }
213.
214.
215.
216.             // Setup inputs
217.             var inputs = new List<NamedOnnxValue>
218.             {
219.                NamedOnnxValue.CreateFromTensor("image", input)
220.             };
221.
222.             // Run inference
223.             using var session = new InferenceSession(modelFilePa
    th);
224.             using IDisposableReadOnlyCollection<DisposableNamedO
    nnxValue> results = session.Run(inputs);
225.
226.             // Postprocess to get softmax vector
227.             IEnumerable<float> output = results.First().AsEnumer
    able<float>();  // First(): The first element in the specified se
    quence. AsEnumerable:
```

```
228.             float sum = output.Sum(x => (float)Math.Exp(x));    /
   / sum(e^x)
229.             IEnumerable<float> softmax = output.Select(x => (flo
   at)Math.Exp(x) / sum);  // e^x / sum
230.
231.             //Extract top 1 predicted classes
232.             IEnumerable<Prediction> top1 = softmax.Select((x, i)
    => new Prediction { Label = LabelMap.Labels[i], Confidence = x }
   )
233.             .OrderByDescending(x => x.Confidence)
234.             .Take(1);
235.             string Tag = "none";
236.             foreach (var t in top1)
237.             {
238.                 string c = t.Confidence.ToString();
239.                 Tag = c;
240.             }
241.
242.             return Tag;
243.         }
244.
245.         private void linkLabel1_LinkClicked(object sender, LinkL
   abelLinkClickedEventArgs e)
246.         {
247.             linkLabel1.LinkVisited = true;
248.             _form1.Show();
249.             this.Hide();
250.         }
251.     }
252.     public class Prediction
253.         {
254.             public string Label { set; get; }
255.             public float Confidence { set; get; }
256.         }
257.         public static class LabelMap
258.         {
259.             static LabelMap()
260.             {
261.                 Labels = new string[]
262.                 {
263.             "新冠患者","病毒性肺炎","正常"
264.                 };
265.             }
266.
```

```
267.             public static string[] Labels { set; get; }
268.
269.
270.
271.         }
272.     public class separation
273.     {
274.         public string Separate()
275.         {
276.             string path = System.AppDomain.CurrentDomain.BaseDir
    ectory;
277.             var files = Directory.GetFiles(path, "*.jpg");
278.             string filen = files[0];
279.             string filename = Path.GetFileName(filen);//张三
    _114514_M.jpg
280.             return filename;
281.         }
282.     }
283.    // private static void pictureBox1_Click(object sender, Event
    Args e)
284.    // {
285.
286.    // }
287.         //private void textBox2_TextChanged(object sender, Event
    Args e)
288.    // {
289.
290.   //  }
291.     }
```