基于 U-Net 的肺炎图像 x-ray 分类与病灶检测

一、研究背景与目的

自从 2019 年底新冠疫情爆发以来，对新冠患者的诊断一直是一个大问题。常见的诊断方法有核酸检测、抗体检测、肺部影像学检测等。其中，肺部的 X 光或 CT 图像一般被认为是确诊的金标准。然而，通过图像的确诊，目前往往需要通过有经验的医生观察而得出结论，无法做到快速与自动化。而机器学习的手段在图像分类，尤其是是否患病的简单二分类问题上表现的很好，但单独的分类问题过于简单，无法符合课程大作业的目标，于是希望设计一个分类与识别病灶的简单网络模型，同时在过程中加入本学期学到的各种图像基本操作手段，通过大作业实现对整个学期课程的融会贯通。

但在寻找之后，发现并没有标注好病灶位置的新冠肺炎数据集，难以训练且难以判断准确率，所以选择对普通肺炎进行训练，学会通过机器学习的手段进行病灶识别分割的基本方法，为之后的进一步学习做好准备。

二、数据集

使用 kaggle 上的”RSNA Pneumonia Detection Challenge”内附带的肺炎数据集，有 30227 个标注好的病人数据与 3000 个未标注的用于预测，网址如下：
https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/data

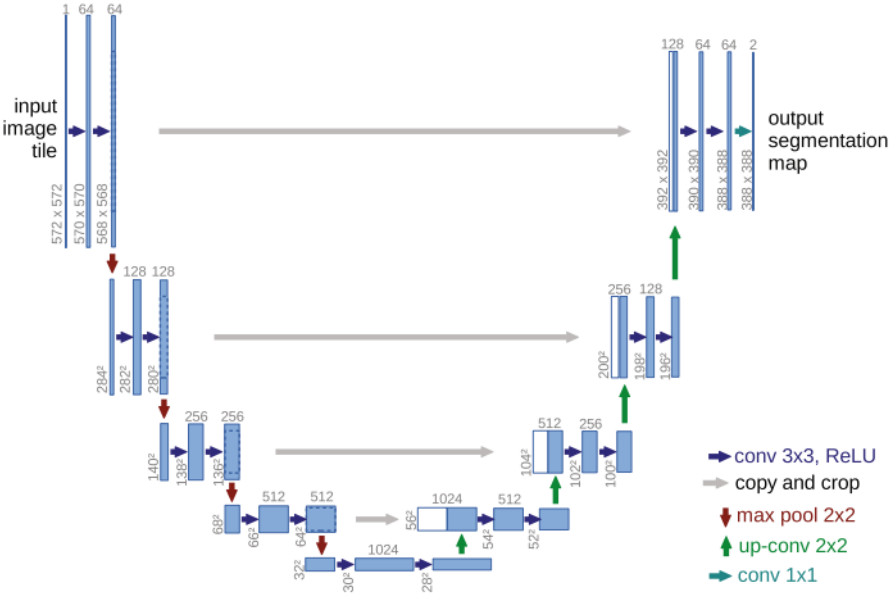
三、算法实现

1、图像预处理

主要对图像大小与灰度范围进行了归一化处理。通过 MinMaxScaler 的方式。

2、U-net 基本结构

采用的是非常常见的医学图像分割算法 U-net。U-net 主要包含两个部分，即特征提取和上采样，网络结构形状类似 U 型，所以叫做 Unet 网络。



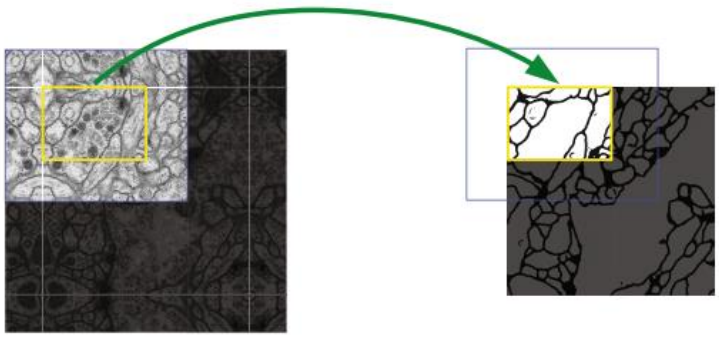
在特征提取部分，每经过一个池化层就有一个尺度，包括原图共有五个尺度。上采样

部分，每上采样以此，就和特征提取部分对应的通道数相同尺度融合，但融合前要先crop。可见输入的 572x572，输出为 388x388。图中每一格蓝色箭头代表 3x3 的卷积操作，且 stride 为 1，padding 为 valid，所以每一次卷积 featuremap 大小减 2。红色箭头代表 2x2 的池化。绿色代表 2x2 的反卷积，将 featuremap 大小乘以 2。灰色箭头表示复制和剪切，是图片大小符合需要。最后则是 1x1 的分类层。
实际具体采用的每层结构如下：

```
PneumoniaUNET(
  (down_1): Sequential(
    (0): conv_block(
      (conv): Conv2d(1, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (bn): BatchNorm2d(64, eps=1e-05, momentum=0.9, affine=True, track_running_stats=True)
      (activ): LeakyReLU(negative_slope=0.03)
    )
    (1): conv_block(
      (conv): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (bn): BatchNorm2d(64, eps=1e-05, momentum=0.9, affine=True, track_running_stats=True)
      (activ): LeakyReLU(negative_slope=0.03)
    )
  (up_5): Sequential(
    (0): conv_block(
      (conv): Conv2d(768, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (bn): BatchNorm2d(512, eps=1e-05, momentum=0.9, affine=True, track_running_stats=True)
      (activ): LeakyReLU(negative_slope=0.03)
    )
    (1): conv_block(
      (conv): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (bn): BatchNorm2d(512, eps=1e-05, momentum=0.9, affine=True, track_running_stats=True)
      (activ): LeakyReLU(negative_slope=0.03)
    )
  )
  (up_5_t): conv_t_block(
    (conv_t): ConvTranspose2d(512, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (bn): BatchNorm2d(256, eps=1e-05, momentum=0.9, affine=True, track_running_stats=True)
    (activ): LeakyReLU(negative_slope=0.03)
  )
```

3、overlap-tile

为防止 overlap 后图像边界的图像块没有周围像素导致信息丢失，文献作者对轴为像素采用镜像扩充。同时为防止第一块图像周围与第二块图像重叠，卷积时采用 valid 卷积和 crop 裁剪，使传到下一层的只有原先黄色部分。



4、弹性变换

为防止过拟合，加入了弹性变换，弹性变化是对像素点各个维度产生（-1，1）区间的随机标准偏差，用高斯滤波对各维度的偏差矩阵进行滤波，最后用放大系数控制偏差范围。因而由 A（x，y）得到的 A'(x+delta_x,y+delta_y)。A'的值通过在原图像差值得到，A'的值充当原来 A 位置上的值。实现代码如下：

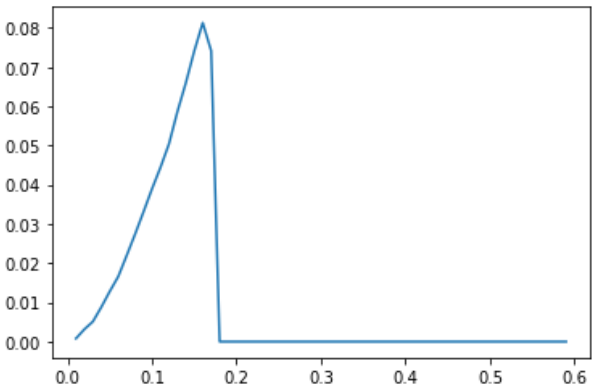
```
def elastic_transform(image, alpha, sigma, random_state=None):
    assert len(image.shape)==2, 'Image must have 2 dimensions.'
    if random_state is None:
        random_state = np.random.RandomState(None)
    shape = image.shape
    dx = gaussian_filter((random_state.rand(*shape) * 2 - 1), sigma, mode="constant", cval=0
* alpha
    dy = gaussian_filter((random_state.rand(*shape) * 2 - 1), sigma, mode="constant", cval=0
* alpha
    x, y = np.meshgrid(np.arange(shape[0]), np.arange(shape[1]), indexing='ij')
    indices = np.reshape(x+dx, (-1, 1)), np.reshape(y+dy, (-1, 1))
    image_warped = map_coordinates(image, indices, order=1).reshape(shape)
    return image_warped
```

5、损失函数

损失函数采用 sigmod+BCELoss 再做交叉熵实现，对每个像素输出分别做 softmax，可以更加注重边缘信息。代码中直接用 BCEWithLogitsLoss2d 函数实现。

6、参数调整

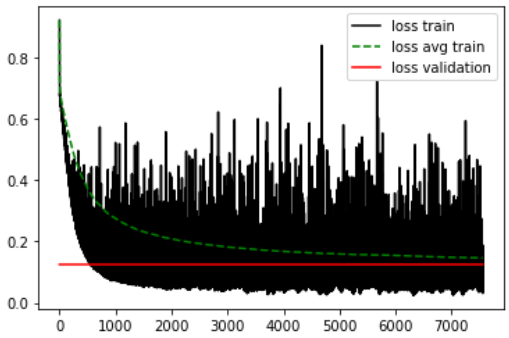
在模型的构建上，主要参考了文献中的网络构建方式，并未进行太大改动。之后主要对 thresholds 进行了超参数搜索：



可以据此找到一个最好的 thresholds。对于其它超参数同理，由于时间与算力关系，在此并未全部进行网格搜索。

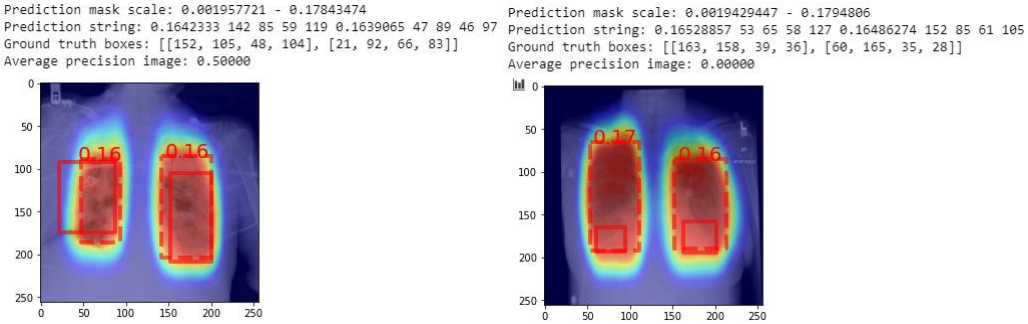
四、结果

1、loss



可以看到对于测试集的 loss 下降较为平滑，最终在 0.15 左右，较为理想。

2、测试集预测



可以看到，部分识别结果较为符合预期，有些结果还是不太理想，但值得注意的是，因为时间关系（数据量较大，训练一个 epoch 需要 90min 左右），所以只进行了一个 epoch 的训练，如果增加 epoch 的量，结果应该会更加符合实际。

五、总结与展望

总的来说，本次大作业实验是对本学期医学图象课程的一次总结，以运用机器学习的图像分割问题为基础，用到了很多课上学到的基本图像处理思想和方法，与实验课更加基础的实验相比，复现了一篇较新的文章，两种作业相互结合，使得医学图像分析处理的综合能力有了较大的提升。

但最终预测结果并不很理想，主要原因是未对数据集进行分割，每次训练的数据太多，训练一个 epoch 就需要 90min 左右（同时由于校园网最近不太稳定，还经常出现训练到一半与服务器连接断开的问题），所以也并未尝试进行更多的超参数搜索与模型参数调整，很多直接采用了文献中或 github 已有模型的数值，但毕竟分类问题不一样，所以结果不好也是正常的，而这点也是机器学习方法所固有的问题。

之后也可以尝试其它如 RCNN 或 yolo 等方法，或与通过传统手段进行的分割结果进行对比，可以对图像分割有一个更好的理解。

在实际应用上，对于新冠肺炎的病灶位置预测可能实际意义并不是很大，但在其它方面，如血管分割等，通过机器学习的手段进行分割的意义还是很大的，之后我们也将继续努力学习这部分的知识，争取真正为临床做出贡献。