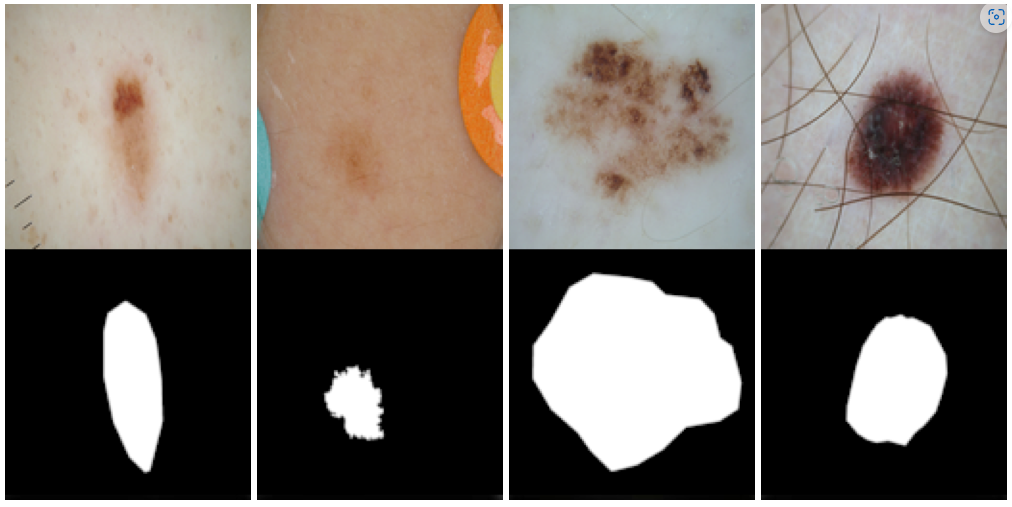
## 数据集说明

皮肤病灶图像 ISIC2018分割任务数据集

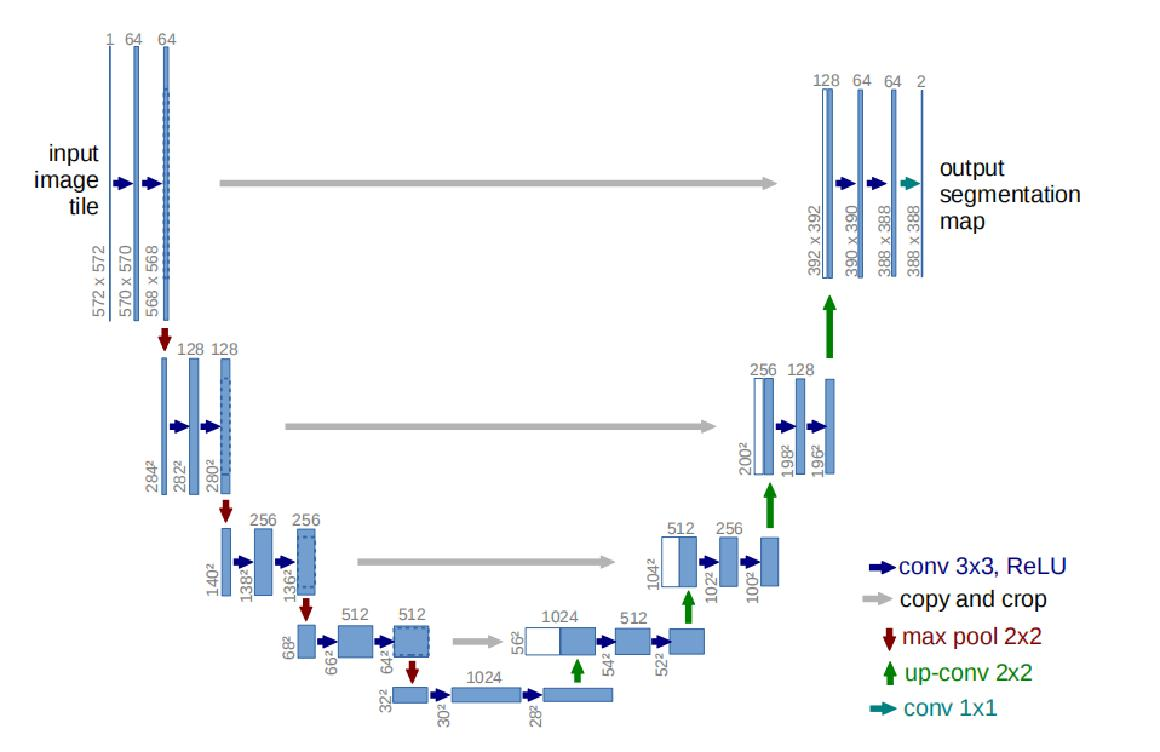
https://challenge.isic-archive.com/landing/2018/45/



[使用Mindspore 加载 Dataset、Dataloader](https://www.mindspore.cn/tutorials/zh-CN/r2.0/beginner/dataset.html#%E8%87%AA%E5%AE%9A%E4%B9%89%E6%95%B0%E6%8D%AE%E9%9B%86)。

由于数据量比较大，算力优先，仅加载部分数据用于训练。

## Unet模型



模型代码：

1. class UNet(nn.Cell):
2. def \_\_init\_\_(self, in\_ch=3, out\_ch=1):
3. super(UNet, self).\_\_init\_\_()
4. *# Down*
5. self.inchannel = DoubleConv(in\_ch, 64)
6. self.down1 = Down(64, 128)
7. self.down2 = Down(128, 256)
8. self.down3 = Down(256, 512)
9. self.down4 = Down(512, 1024)
10. *# UP*
11. self.up1 = Up(1024+512, 512)
12. self.up2 = Up(512+256, 256)
13. self.up3 = Up(256+128, 128)
14. self.up4 = Up(128+64, 64)
15. self.final = nn.Conv2d(64, out\_ch, kernel\_size=1)  *# (64 1 1) out每个像素点是一个概率值，类似2分类*
17. *# output*
18. self.sigmoid = ops.Sigmoid()
19. def construct(self, x):
20. x1 = self.inchannel(x)
21. x2 = self.down1(x1)
22. x3 = self.down2(x2)
23. x4 = self.down3(x3)
24. x5 = self.down4(x4)
25. x = self.up1(x5, x4)
26. x = self.up2(x, x3)
27. x = self.up3(x, x2)
28. x = self.up4(x, x1)
29. x = self.final(x)
30. output = self.sigmoid(x)
31. return output

## 评估指标

在二分类问题中，TP、FP和FN是衡量模型分类性能的三个指标。

- TP（True Positive）表示真正例，即模型正确地预测为正例的样本数。

- FP（False Positive）表示假正例，即模型错误地预测为正例的样本数。

- FN（False Negative）表示假负例，即模型错误地预测为负例的样本数。

在图像分割问题中，可以将TP、FP和FN应用于像素级别的分割结果。

除常规的Accuracy外，还有图像分割常用的评估指标：IoU，Dice系数等

IoU指标测量预测分割结果与真实分割结果之间的重叠程度，其计算公式为：

IoU = TP / (TP + FP + FN) 即\*\*交集/并集\*\*

Dice系数也是一种常用的衡量图像分割模型性能的指标，其计算公式为：

Dice = (2 \* TP) / (2 \* TP + FP + FN)，与交并比类似，便于求值。

## 模型训练

训练日志全部记载在**train.log** 文件中，eg：

Epoch: **[** 8**/**200**]**, Steps: **[** 33**/** 40**]**, Train Loss: **[**0.6737793684**]**

Epoch: **[** 8**/**200**]**, Steps: **[** 34**/** 40**]**, Train Loss: **[**0.6876214743**]**

Epoch: **[** 8**/**200**]**, Steps: **[** 35**/** 40**]**, Train Loss: **[**0.7317028046**]**

Epoch: **[** 8**/**200**]**, Steps: **[** 36**/** 40**]**, Train Loss: **[**0.6832416654**]**

Epoch: **[** 8**/**200**]**, Steps: **[** 37**/** 40**]**, Train Loss: **[**0.6824316978**]**

Epoch: **[** 8**/**200**]**, Steps: **[** 38**/** 40**]**, Train Loss: **[**0.6534892917**]**

Epoch: **[** 8**/**200**]**, Steps: **[** 39**/** 40**]**, Train Loss: **[**0.6900650859**]**

Epoch: **[** 8**/**200**]**, Steps: **[** 40**/** 40**]**, Train Loss: **[**0.6673911810**]**

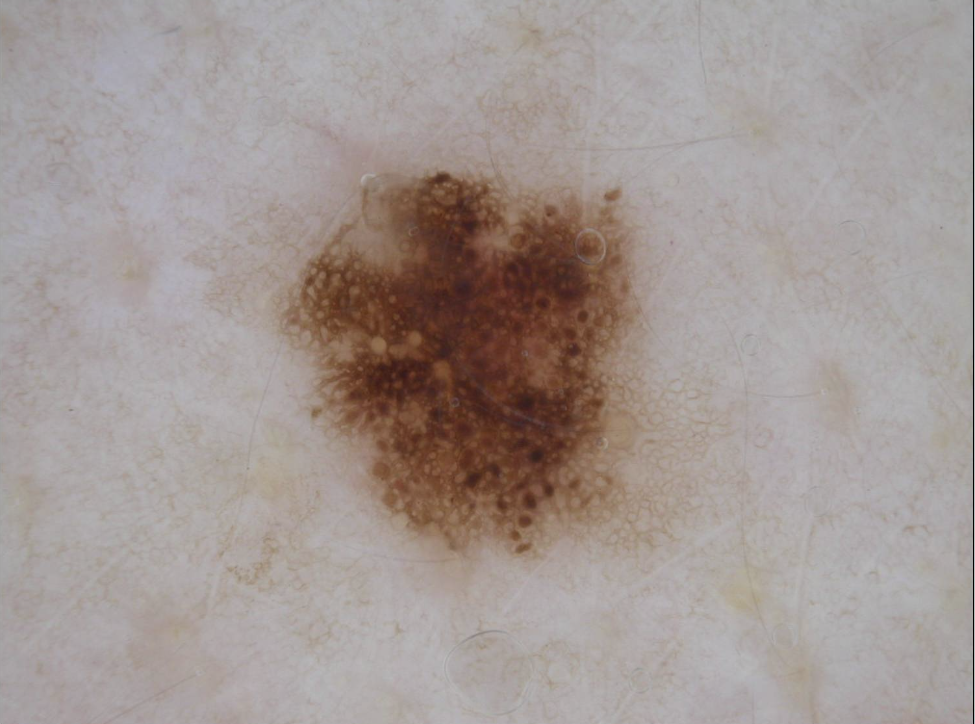
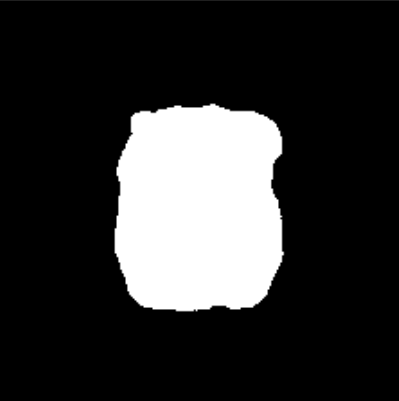
--------------------------------------------------

Epoch: **[** 8**/**200**]**, Average Train Loss: **[**0.6905487776**]**

Val loss:0.715175 丨acc: 0.783丨丨iou: 0.381丨丨dice: 0.549丨丨sens: 0.532丨丨spec: 0.893丨

## 模型预测

运行predict.py 代码，加载训练好的模型，

## 总结

第一次使用mindspore框架，在训练时优化器选择，损失，以及整个训练过程比较保守，所以效果欠佳，不好调试。

另外，最重要的是面对新的框架，官方文档是最重要的资料，帮助我解决了许多问题，包括自定义数据集加载，模型训练环境设置等。