同济大学
TONGJI UNIVERSITY

人工智能基础课程报告
——冠脉介入手术智能规划系统

Report on Fundamentals of Artificial Intelligence Course —— Intelligent Percutaneous Coronary Intervention Guiding System

学　　院　电子与信息工程学院

专　　业　自动化

组　　号　第 21 组

组长姓名　姚天亮　学　　号　2150248

组员姓名　瞿闻希　学　　号　2152954

组员姓名　张旭晖　学　　号　2151399

组员姓名　张嘉诚　学　　号　2150251

指导教师　尹慧琳

2023 年 12 月 27 日

目 录

一、背景介绍

冠状动脉介入治疗，特别是经皮冠状动脉介入治疗（PCI），开创了冠状动脉疾病（CAD）治疗的新时代。CAD是一种常见的心血管疾病，其特征是在冠状动脉内形成动脉粥样硬化斑块，导致流向心脏的血流量减少。PCI是一种基于导管、导丝的方法，用于扩张狭窄或阻塞的冠状动脉。虽然PCI在恢复血流方面取得了非凡的成功，但它有临床局限性。

血管导丝常用于介入手术过程中，如PCI。这些设备需要医生精密的操作，在介入手术[1]中需要高度的手眼协调。同时，在实际手术过程中，心内科医生仅限于基于二维（2D）血管造影来评估血管导丝的位置。这就需要对心内科医生进行充分的专业培训和实践，以确保精准的介入操作[2]。然而，由于血管结构的复杂性，即使是最有经验的心内科医生也可能在确定最佳介入途径方面面临挑战。为了减轻这些挑战，人们提出了实施计算机辅助介入手术系统。这种系统的潜在好处如下：[1]：

(1) 提高介入准确性[4]：计算机辅助介入路径规划可以通过利用个性化图像数据和患者特异性解剖结构，精确规划最佳介入路径。

(2) 提高介入效率[5]：计算机辅助的自动介入路径规划有助于快速生成最佳的介入路径，显著减少了心内科医生在手术过程中的决策时间。

(3) 提升手术自动化[6]：将计算机辅助的自动介入路径规划和引导集成到介入机器人系统中，通过自动跟踪生成的介入路径，提高当前介入机器人的自主性。

心内科医生在PCI手术期间使用X光成像，并引导定期注射小剂量的造影剂得到2D数字减影血管造影（DSA）图片来查看即时的手术环境。因此，心内科医生缺乏对完整循环系统的全面视角，只能实时看到局部血管成像的动态变化。因此，开发实时局部路径规划技术对PCI至关重要。目前，PCI领域介入导航主要可分为两大类：基于血流动力学参数的方法和基于医学图像的方法。然而，基于血流动力学参数的方法需要使用传感器来检测血流动力学数据，这使得心内科医生的介入程序复杂化，且无法直观为心内科医生在PCI期间展示应该采取的路径。与此同时，现有基于成像的方法，如血管内超声（IVUS），缺乏像我们日常生活中导航系统等直观的引导。

为了让心脏病专家对实际的介入情况进行更精确和直观的评估，本项目中提出了全新的PCI引导系统使用了DSA图像分割。这就实现了可靠的导航支持技术。基于DSA图像的二维路径规划算法和血管DSA图像分割网络构成了PCI引导系统。为了提高血管DSA图像分割的精度，提出了一种基于最先进的ResUNet++架构的增强模型，并结合了坐标注意机制。该模型使用Softdice Loss损失函数进行训练和优化。此外，对DSA数据采用了基于RetinexNet的对比度增强预处理技术。此外，还实现了一种基于DSA分割结果特征的实时二维路径规划算法。

二、系统设计

2.1 系统架构

PCI 指导系统所提出的框架工作流程如图 1 所示。DSA 图像通过一个语义分割模块进行处理。在该语义分割模块中，从 DSA 图像序列中提取帧，并进行对比度增强进行预处理。随后，引入了改进的 ResUNet++网络来分割血管轮廓。在路径规划模块中，该算法基于分割结果和选择点生成 PCI 的局部最优和参考路径。

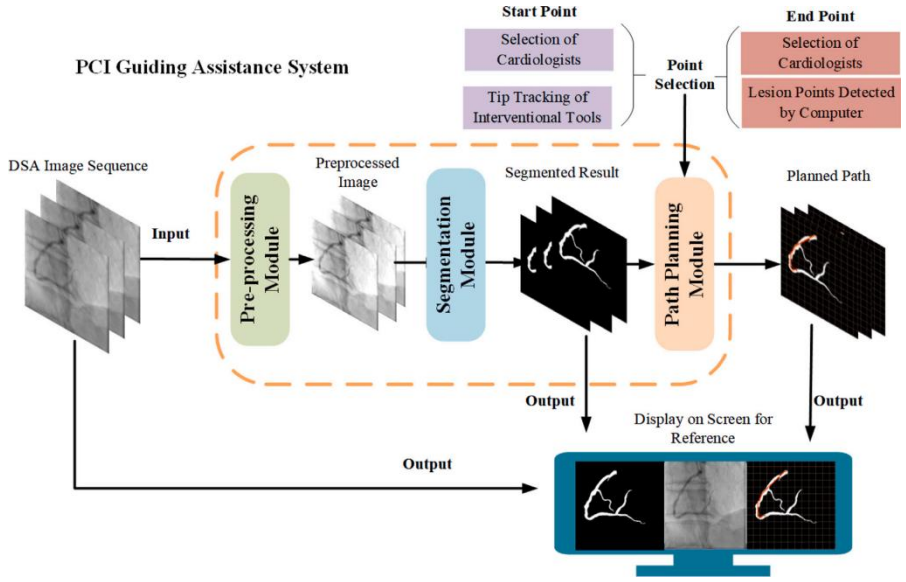


图 1. PCI 指导辅助系统的架构。该系统由两个主要模块组成：(1)语义分割模块，利用改进的 ResUNet++网络从 DSA 图像中提取冠状动脉血管系统。单个 DSA 帧首先进行对比度增强预处理，然后输入分割模型。(2)路径规划模块，根据分割结果和临床医生指示的选定目标点，自动生成最优的、解剖学上可行的 PCI 干预路径。整个框架旨在为经皮冠状动脉手术提供实时可视化和决策支持。

2.2 语义分割模型

ResUNet++是ResUNet的一个高级版本，它包含了挤压和激发（SE）模块[25]，改进的网络还引入了空洞卷积空间金字塔池（ASPP）模块取代了原来的桥接模块，并将自注意机制嵌入到网络中。SE 可以自适应地调整特征的信道响应特性，增强了网络的特征表示能力。ASPP 通过使用不同采样率的多个并行扩张卷积层对特征进行重采样，提高了空间特征表示能力。因此，通过将这些模块引入网络，该模型可以从不同尺度上获取更丰富、更准确的特征信息，从而提高语义分割的准确性和鲁棒性。解码模块由三组解码器块层组成，它们对特征图进行上采样，同时，ASPP 使用多个并行扩展卷积捕获多尺度上下文信息。分割结果是通过通过输出块中的卷积层合并上采样的特征映射而产生的。

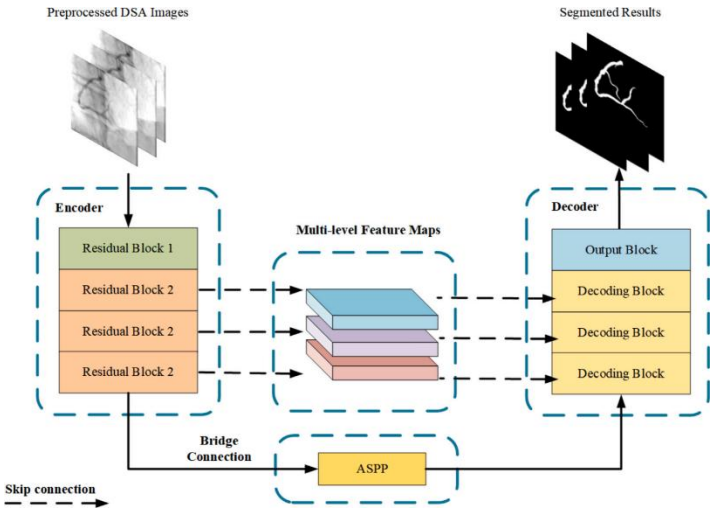


图 2. 所提出的分割模型架构的说明。该模型由三个模块组成：编码模块（左）用于利用具有坐标注意的残差块从输入图像中提取多层特征；桥连接模块（中）用于通过 ASPP 构建层次特征；解码模块（右）用于上采样特征并生成具有多个扩展卷积的分割结果。

各种类型的损失函数可以应用于不同的图像分割任务。每个损失函数在表示分割区域的属性，解决类的不平衡，并量化预测的和 Ground Truth[26]之间的一致性。Softdice Loss 是一种适用于医学图像分割的损失函数，可以有效地解决语义分割中的类不平衡问题。其优点在于，它不仅考虑了每个像素的分类精度，而且强调了分割结果与地面真实标签之间的重叠程度。对于二维冠状动脉分割，SoftDice 损失功能可以有效地解决血管边界模糊的问题。这一优势不仅提高了一般目标的分割性能和稳定性，而且也提高了具有复杂形状的目标，而其他方法往往表现出较差的性能。

$$L_{SoftDice} = 1 - \frac{2\sum_{i=1}^{N} y_i p_i + \epsilon}{\sum_{i=1}^{N} y_i + \sum_{i=1}^{N} p_i + \epsilon}$$

2.3 路径规划算法

提出了一种基于启发式搜索的介入路径规划算法，该算法可以在复杂环境中找到从起点到终点的最优路径。启发式路径规划算法可以在有限的时间内找到次优或接近最优的路径，适用于实时或动态路径规划问题。因此，我们采用了一种启发式的方法来进行路径规划。在算法中，分割结果划分为网格图，其中每个网格单元代表一个节点。

首先，根据医学图像选择起点和端点，系统将 DSA 图像的分割结果分为网格图，每个节点表示可步行或非可步行状态，以及到端点的距离估计值。然后，该算法从起点开始搜索可行走的节点，根据每个网格单元的实际距离和距离估计值计算一个启发式函数和代价函数，并将具有最小代价函数值的节点添加到优先级队列中。接下来，算法不断从优先队列中取出代价最小函数值的节点，并扩展其相邻的可行走节点，直到遇到边界或端点。

Algorithm 1 Path planning for PCI

Divide the 2D DSA images into a grid map of size 256*256, and each grid can be considered as a node. The node domain is $S = \{S_1, S_2, \ldots, S_n\}$.
Initialize the open list and the closed list.
Set a start node S_1 and a goal node S_n.
Calculate the cost function $f(n)$, defined as $f(n) = g(n) + h(n)$ of S_1. Set the $g(S_1) = 0$, $h(S_1) = (S_{1x} - S_{nx})^2 + (S_{1y} - S_{ny})^2$, $f(S_1) = g(S_1) + h(S_1)$.
Put S_1 into the open list.
while the open list is not empty **do**
    Generate the neighbors of the current node S_i by moving in eight directions, the domain of neighbour nodes is $P = \{P_1, P_2, \ldots, P_8\}$.
    for each P_i **do**
        if P_i = *one of parent node*. Delete P_i.
        Elseif P_i is the border: Delete P_i.
        Else Calculate the value of f.
            if P_i with the lowest f value from the open list: Add it to the closed list and check if $P_i == S_n$.
                if $P_i == S_n$: Return the path by tracing back the parent nodes.
                Else continue.
            Else continue.
    end of for
    end of while

数据集包括冠状动脉疾病的 DSA 图像。在数据集中，每张 DSA 图像的分辨率为 512×512 像素和一个通道。该训练集包括来自 192 名患者的 768 个 mask 和 768 张冠状动脉造影。验证集包含来自 73 名患者的 200 个 mask 和 200 个冠状动脉造影。该测试集包括来自 23 名患者的 109 个独立的冠状动脉 DSA。它们是由经验丰富的高年级医学院心内科同学进行注释的。需要指出的是：训练集、验证集和测试集没有重叠的患者。

三、实验结果

为了评估所提出的分割模型的性能，并将其与其他基线方法进行比较，我们利用了四个指标：精度、F1 分数、Jaccard 指数和准确性。这些指标是计算机视觉中最常用的指标，通过测量预测的分割结果和 Ground Truth 真实注释之间的相似性来评估模型的性能。

$$Precision = \frac{TP}{TP + FP}$$

$$F1 = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Jaccard = \frac{TP}{TP + FP + FN}$$

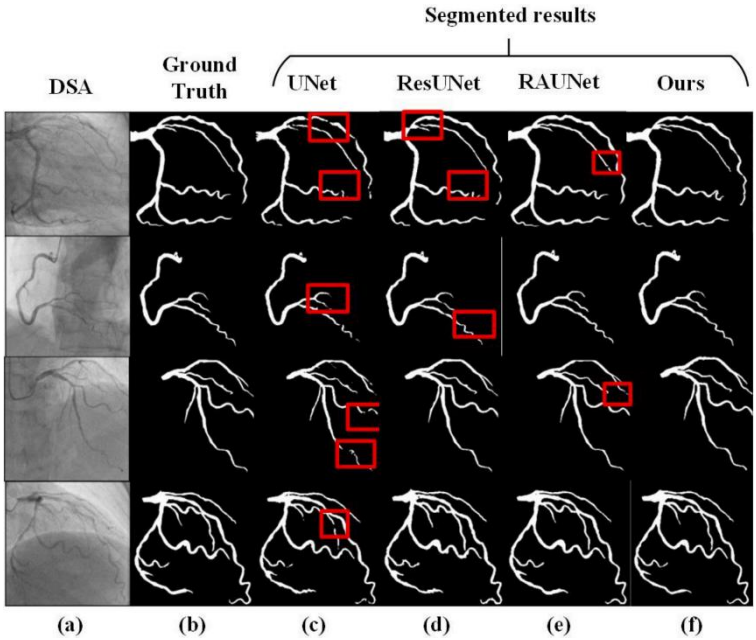


图 3. 四种可视化方法的血管分割结果。(a)原始的 DSA 图像。(b)Ground Truth 标签。(c)的分割结果来自 UNet。(d)是来自 ResUNet 的分割结果。(e)来自 RAUNet 的分割结果。(f)来自我们提出的模型的分割结果。红框中的结构是模型无法识别的微小细节。该图验证了我们的模型在 DSA 图像上识别复杂的冠状动脉血管的有效性。

表 1. 分割性能比较

Model	Jaccard	F1	Precision	Acc.
UNet	0.2604	0.2134	0.5604	0.8304
ResUNet	0.3271	0.3825	0.6458	0.9125
RAUNet34	0.3550	0.4006	**0.7425**	0.9419
TransResUNet	**0.4506**	**0.6167**	0.5579	0.9439
Ours	0.4091	0.5626	0.7334	**0.9583**

Model	Jaccard	F1	Recall	Precision	Acc.
Trans ResUNet	0.557	0.710	0.632	**0.828**	0.968
Ours	**0.611**	**0.755**	**0.709**	0.815	**0.971**

表 2. 实验硬件设置

Experimental setup.

Module	Version
CPU	Intel(R) Core(TM) i7-10870H, 2.20GHz
Memory	8.00 GB
GPU	GeForce GTX1660 Ti
System	Windows 10
CUDA Version	12.0
Pytorch Version	1.13.1

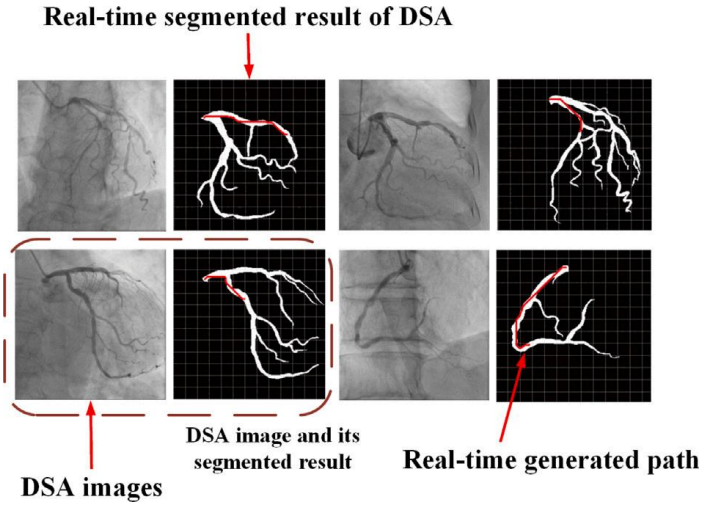


图 4. 本生成的冠状动脉介入治疗路径可视化临床 DSA 图像。

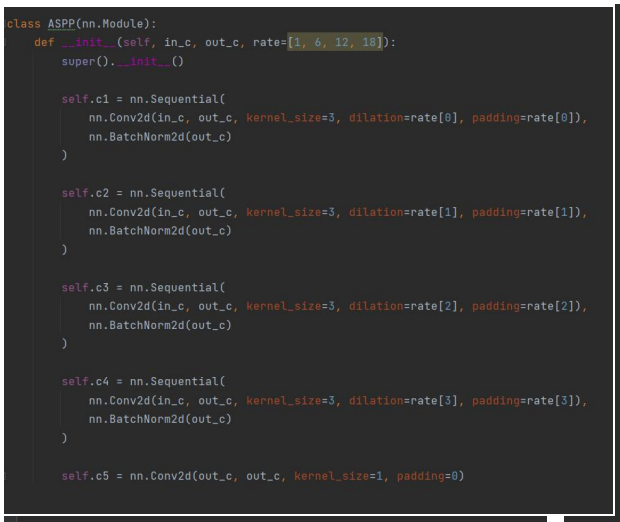
四、代码实现

```python
lr = 1e-4
nums_layer = 5
load_from_check_point = False  # set false to train from the scratch; otherwise set iter num to resume training

self.net = RAUNet34.RAUNet().to(self.device)
# self.net = unet.UNet().to(self.device)
# self.net = ResNet.ResNet18().to(self.device)
# self.net = ResUNet.ResUnet(3).to(self.device)
# self.net = ResUnetPPP.ResUNetPP(nums_layer).to(self.device)
# self.net = ResNeXt.resnext50().to(self.device)
# self.net = ResUnetPPP.RetinexNetResUNetPP(nums_layer).to(self.device)
# self.net = ResUNet.ResUnet(3).to(self.device)
# self.net = TransResUNet.TransResUNet().to(self.device)
# self.net = mobilenetv2.mobilenetv2().to(self.device)

# 优化器
self.opt = torch.optim.Adam(self.net.parameters())

# 这里直接使用结合Sigmoid的二分类交叉熵来训练
# 可以尝试改其他损失，DiceLoss、FocalLoss
self.loss_func = nn.BCEWithLogitsLoss()
self.dice_func = segmentation_loss.SoftDiceLoss()
```

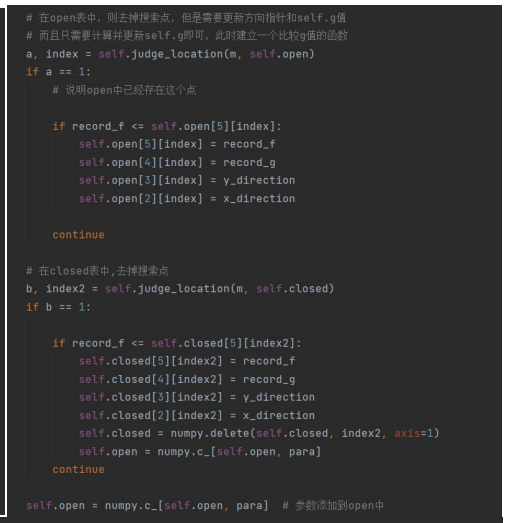
```python
class Squeeze_Excitation(nn.Module):
    def __init__(self, channel, r=8):
        super().__init__()

        self.pool = nn.AdaptiveAvgPool2d(1)
        self.net = nn.Sequential(
            nn.Linear(channel, channel // r, bias=False),
            nn.ReLU(inplace=True),
            nn.Linear(channel // r, channel, bias=False),
            nn.Sigmoid(),
        )

    def forward(self, inputs):
        b, c, _, _ = inputs.shape
        x = self.pool(inputs).view(b, c)
        x = self.net(x).view(b, c, 1, 1)
        x = inputs * x
        return x
```

```python
def h_value_tem(self, son_p):
    """
    计算拓展节点和终点的h值
    :param son_p:子搜索节点坐标
    :return:
    """
    h = (son_p[0] - self.goal[0]) ** 2 + (son_p[1] - self.goal[1]) **2
    h = numpy.sqrt(h)  # 计算h
    return h
```

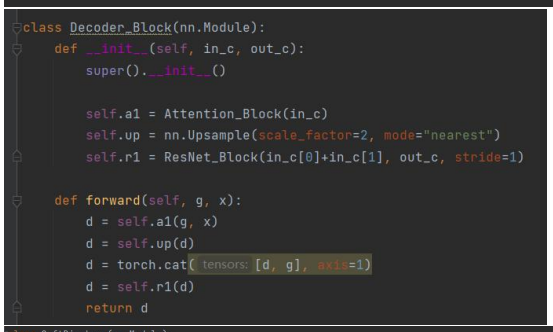
```python
def draw_init_map(self):
    """
    画出起点终点图
    :return:
    """
    plt.imshow(map_grid, cmap=plt.cm.hot, interpolation='nearest', vmin=0, vmax=10)
    # plt.colorbar()
    xlim(*args -1, map_lon)  # 设置x轴范围
    ylim(*args -1, map_lat)  # 设置y轴范围
    my_x_ticks = numpy.arange(start 0, *args map_lon, 20)
    my_y_ticks = numpy.arange(start 0, *args map_lat, 20)
    plt.xticks(my_x_ticks)
    plt.yticks(my_y_ticks)
    plt.grid(True)
    # plt.show()
```



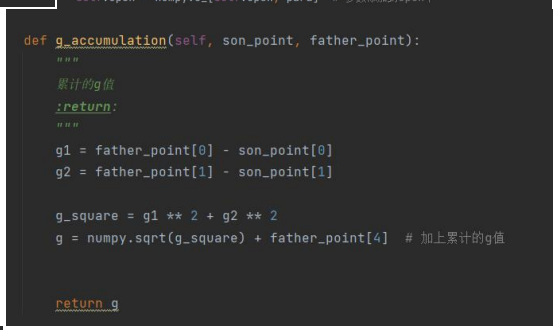
```python
class ASPP(nn.Module):
    def __init__(self, in_c, out_c, rate=[1, 6, 12, 18]):
        super().__init__()

        self.c1 = nn.Sequential(
            nn.Conv2d(in_c, out_c, kernel_size=3, dilation=rate[0], padding=rate[0]),
            nn.BatchNorm2d(out_c)
        )

        self.c2 = nn.Sequential(
            nn.Conv2d(in_c, out_c, kernel_size=3, dilation=rate[1], padding=rate[1]),
            nn.BatchNorm2d(out_c)
        )

        self.c3 = nn.Sequential(
            nn.Conv2d(in_c, out_c, kernel_size=3, dilation=rate[2], padding=rate[2]),
            nn.BatchNorm2d(out_c)
        )

        self.c4 = nn.Sequential(
            nn.Conv2d(in_c, out_c, kernel_size=3, dilation=rate[3], padding=rate[3]),
            nn.BatchNorm2d(out_c)
        )

        self.c5 = nn.Conv2d(out_c, out_c, kernel_size=1, padding=0)
```



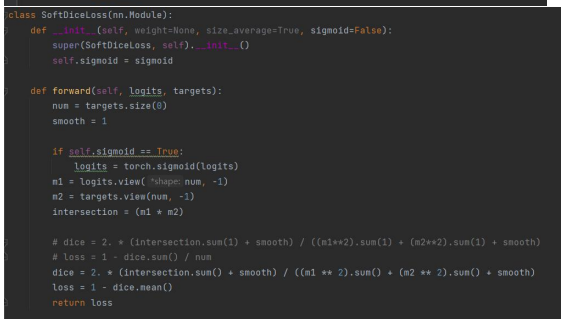
```python
# 在open表中，则去掉搜索点，但是需要更新方向指针和self.g值
# 而且只需要计算并更新self.g即可，此时建立一个比较g值的函数
a, index = self.judge_location(m, self.open)
if a == 1:
    # 说明open中已经存在这个点

    if record_f <= self.open[5][index]:
        self.open[5][index] = record_f
        self.open[4][index] = record_g
        self.open[3][index] = y_direction
        self.open[2][index] = x_direction

    continue

# 在closed表中,去掉搜索点
b, index2 = self.judge_location(m, self.closed)
if b == 1:

    if record_f <= self.closed[5][index2]:
        self.closed[5][index2] = record_f
        self.closed[4][index2] = record_g
        self.closed[3][index2] = y_direction
        self.closed[2][index2] = x_direction
        self.closed = numpy.delete(self.closed, index2, axis=1)
        self.open = numpy.c_[self.open, para]
    continue

self.open = numpy.c_[self.open, para]  # 参数添加到open中
```



```python
class Decoder_Block(nn.Module):
    def __init__(self, in_c, out_c):
        super().__init__()

        self.a1 = Attention_Block(in_c)
        self.up = nn.Upsample(scale_factor=2, mode="nearest")
        self.r1 = ResNet_Block(in_c[0]+in_c[1], out_c, stride=1)

    def forward(self, g, x):
        d = self.a1(g, x)
        d = self.up(d)
        d = torch.cat( tensors: [d, g], axis=1)
        d = self.r1(d)
        return d
```



```python
def g_accumulation(self, son_point, father_point):
    """
    累计的g值
    :return:
    """
    g1 = father_point[0] - son_point[0]
    g2 = father_point[1] - son_point[1]

    g_square = g1 ** 2 + g2 ** 2
    g = numpy.sqrt(g_square) + father_point[4]  # 加上累计的g值


    return g
```



```python
class SoftDiceLoss(nn.Module):
    def __init__(self, weight=None, size_average=True, sigmoid=False):
        super(SoftDiceLoss, self).__init__()
        self.sigmoid = sigmoid

    def forward(self, logits, targets):
        num = targets.size(0)
        smooth = 1

        if self.sigmoid == True:
            logits = torch.sigmoid(logits)
        m1 = logits.view( *shape: num, -1)
        m2 = targets.view(num, -1)
        intersection = (m1 * m2)

        # dice = 2. * (intersection.sum(1) + smooth) / ((m1**2).sum(1) + (m2**2).sum(1) + smooth)
        # loss = 1 - dice.sum() / num
        dice = 2. * (intersection.sum() + smooth) / ((m1 ** 2).sum() + (m2 ** 2).sum() + smooth)
        loss = 1 - dice.mean()
        return loss
```

五、小组分工

姚天亮	语义分割算法设计、语义分割实验、报告书写
瞿闻希	语义分割实验、路径规划测试、报告书写
张旭晖	语义分割实验、路径规划算法设计、报告书写
张嘉诚	路径规划算法设计、路径规划测试、报告书写

六、致谢

感谢同济大学医学院 2019 级本科生、北京协和医学院/中国医学科学院阜外医院心内科汪欣怡同学提供的宝贵数据集与医学方面的精心指导和宝贵意见。

参考文献

[1] L. Da, D. Zhang, T. Wang, Overview of the vascular interventional robot, Int. J. Med. Robot. Comput. Assist. Surg. 4 (4) (2008) 289–294, http://dx.doi.org/10. 1002/rcs.212.

[2] S. Guo, J. Cui, Y. Zhao, Y. Wang, Y. Ma, W. Gao, G. Mao, S. Hong, Machine learning-based operation skills assessment with vascular difficulty index for vascular intervention surgery, Med. Biol. Eng. Comput. 58 (2020) 1707–1721, http://dx.doi.org/10.1007/s11517-020-02195-9.

[3] L. Adhami, È. Coste-Manière, Optimal planning for minimally invasive surgical robots, IEEE Trans. Robot. Autom. 19 (5) (2003) 854–863, http://dx.doi.org/10. 1109/TRA.2003.817061.

[4] J. Guo, X. Jin, S. Guo, Study of the operational safety of a vascular interventional surgical robotic system, Micromachines 9 (3) (2018) 119, http://dx.doi.org/10. 3390/mi9030119.

[5] C. Meng, J. Zhang, D. Liu, B. Liu, F. Zhou, A remote-controlled vascular interventional robot: system structure and image guidance, Int. J. Med. Robot. Comput. Assist. Surg. 9 (2) (2013) 230–239, http://dx.doi.org/10.1002/rcs.306.

[6] G.-Z. Yang, J. Cambias, K. Cleary, E. Daimler, J. Drake, P.E. Dupont, N. Hata, P. Kazanzides, S. Martel, R.V. Patel, et al., Medical robotics—Regulatory, ethical, and legal considerations for increasing levels of autonomy, Science Robotics 2 (4) (2017) eaam8638, http://dx.doi.org/10.1126/scirobotics.aam8638.