**东北大学**

**综合实训报告**

接受实习单位: 东北大学实验室

实习日期: 2021年5月14日~2021年6月15日

计算机学院 人工智能 专业 2019 年级 1 班

姓名： 穆奕博 学号 20195205

实习分配班级: 医学影像智能分析班

实习类型：□金工实习□认识实习√生产实习□毕业实习

**实习报告**撰写的内容与要求

1. 实习任务：介绍实习的目的、意义、任务及实习单位的概况等内容。通常以前言或引言形式表述，不单列标题及序号。
2. 实习内容：先介绍实习安排概况，包括时间、地点、内容等，然后逐项介绍具体实习流程与实习工作内容，以及[专业知识](http://www.so.com/s?q=%E4%B8%93%E4%B8%9A%E7%9F%A5%E8%AF%86&ie=utf-8&src=wenda_link)与专业技能在实习过程中的应用。本部分内容应以记叙或[白描手法](http://www.so.com/s?q=%E7%99%BD%E6%8F%8F%E6%89%8B%E6%B3%95&ie=utf-8&src=wenda_link)为基调，在完整叙述的基础上，对自己认为有重要意义或需要研究解决的问题进行重点叙述，其它内容则可简述。
3. 实习结果：围绕实习任务要求，对实习中发现的问题进行分析、思考，提出解决问题的对策、建议等。分析问题、解决问题要有依据（如有[参考文献](http://www.so.com/s?q=%E5%8F%82%E8%80%83%E6%96%87%E7%8C%AE&ie=utf-8&src=wenda_link" \t "_blank)可在正文后附录）。分析讨论的内容、推理过程及所提出的对策与建议作为实习报告的重要内容之一，是反映或评价[实习报告](http://www.so.com/s?q=%E5%AE%9E%E4%B9%A0%E6%8A%A5%E5%91%8A&ie=utf-8&src=wenda_link)水平的重要依据。
4. 实习总结或体会：对实习效果进行综合评价，着重介绍自身的收获与体会，内容较多时可列出[小标题](http://www.so.com/s?q=%E5%B0%8F%E6%A0%87%E9%A2%98&ie=utf-8&src=wenda_link)，逐一列举。总结或体会的最后部分，应针对实习中发现的自身不足，简要地提出今后学习，努力的方向。
5. 将实习日记按照时间顺序以附件形式放在实习报告正文后面。
6. 实习报告正文一律采用计算机排版、A4纸**双面**打印，正文字体为**小四号**宋体，**1.35倍**行距，正文页数不低于15页，一般不超过35页（其中专家报告一共不超过3页，**中文字数**不少于5000字）。每章标题为小三号黑体字居中。全文要求语句通顺、论述严谨规范。
7. 请注意封面页、扉页、评语页等的打印及装订顺序。分别为①封面页（其背面为扉页“实习报告的内容与要求”）、②目录、③正文、④附录-实习日记、⑤其他附件（如校外实习单位的鉴定材料、专利、获奖证书等）、⑥评语页。

**目 录**

[1. 引言 1](#_Toc10364)

[1.1 实习目的和要求 1](#_Toc399)

[1.2 实习内容 2](#_Toc27936)

[1.3 实习单位概况 2](#_Toc24830)

[2. 眼底图像病灶检测及辅助疾病诊断 4](#_Toc20703)

[2.1 任务描述 4](#_Toc4414)

[2.1.1 任务背景 4](#_Toc2252)

[2.1.2 相关工作 4](#_Toc32279)

[2.1.3 任务描述 4](#_Toc14467)

[2.2 算法总体设计 4](#_Toc28612)

[2.3 数据处理 5](#_Toc13860)

[2.3.1 数据来源 5](#_Toc32651)

[2.3.2 数据预处理 5](#_Toc27380)

[2.3.3 数据增强 6](#_Toc14922)

[2.4 模型选择 6](#_Toc7748)

[2.5 模型训练 7](#_Toc10327)

[2.5.1 迁移学习 7](#_Toc9290)

[2.5.2 模型预训练 7](#_Toc27997)

[2.5.3 模型训练 9](#_Toc30994)

[2.6 实验分析 9](#_Toc2075)

[2.6.1 不同数据集的迁移结果讨论 9](#_Toc20406)

[2.6.2 训练结果分析 9](#_Toc14646)

[2.6.3 问题探讨 10](#_Toc8306)

[3. 心室分割与疾病分类 11](#_Toc4474)

[3.1 任务描述 11](#_Toc21594)

[3.1.1 任务背景 11](#_Toc27454)

[3.1.2 任务描述 11](#_Toc25529)

[3.2 算法总体设计 11](#_Toc9310)

[3.3 数据处理 11](#_Toc4305)

[3.3.1 数据来源 11](#_Toc12133)

[3.3.2 数据预处理 12](#_Toc24530)

[3.3.3 数据增强 13](#_Toc16960)

[3.4 模型选择 13](#_Toc14146)

[3.5 模型训练 14](#_Toc23819)

[3.5.1 语义分割 14](#_Toc4146)

[3.5.2 疾病图像分类 15](#_Toc12395)

[3.6 实验分析 16](#_Toc16272)

[3.6.1 训练结果分析 16](#_Toc19589)

[3.6.2 问题探讨 16](#_Toc32524)

[4. 基于CycleGAN的低剂量CT质量优化 17](#_Toc26808)

[4.1 任务描述 17](#_Toc16284)

[4.1.1 任务背景 17](#_Toc30756)

[4.1.2 研究现状 17](#_Toc27681)

[4.1.3 任务描述 17](#_Toc30683)

[4.2 算法总体设计 17](#_Toc7244)

[4.3 数据处理 18](#_Toc7178)

[4.3.1 数据来源 18](#_Toc18898)

[4.3.2 数据预处理 18](#_Toc7394)

[4.4 模型选择 19](#_Toc9255)

[4.5 模型训练 20](#_Toc3054)

[4.5.1 模型训练 20](#_Toc7941)

[4.5.2 损失函数 21](#_Toc31982)

[4.5.3 训练结果分析 21](#_Toc31316)

[4.5.4 问题探讨 22](#_Toc26287)

[5. 实习总结 24](#_Toc13869)

[5.1 实习内容的复杂性评价 24](#_Toc885)

[5.2 实习体会、收获与建议 24](#_Toc15370)

[6. 参考文献 26](#_Toc30197)

[附录 实习日记 27](#_Toc3682)

# 引言

## 实习目的和要求

**1.1.1实习目的**

* 1. 通过深度学习在计算机视觉和医学影像领域的应用，验证和扩展本科前三年所学相关课程的理论知识，在实践中深入理解其中的重点和难点，提高基本理论水平，巩固基础知识体系，掌握计算机系统设计与开发的基本技能，培养灵活运用理论知识分析和解决实际问题的能力。
  2. 通过团队合作开发，培养团结协作的能力。
  3. 通过专家讲座，了解本学科的科学研究前沿及进展，拓宽知识面，拓展学术和国际视野，培养创新精神。
  4. 通过到企业参观与交流，学习课堂以外的实践知识，了解所学专业在社会经济建设中的地位、作用和发展趋势，以及社会对本专业毕业生的素质要求，为最后一学年的课程学习和毕业设计打好基础。
  5. 通过撰写设计需求文档、实习报告和答辩文档，培养学生查阅中英文文献的能力、锻炼口头和书面的表达能力和沟通能力。

**1.1.2 实习要求**

**一般要求**

* 1. 在掌握计算机基本程序设计语言的基础上，利用所学的数学知识、自然科学知识、工程基础知识以及计算机专业知识对所要开发的系统进行软硬件分析和建模。
  2. 在系统的分析与设计过程中，要多方面查阅资料，并利用形式化方法表达所开发系统中涉及的工程元素，采用适当的抽象模型予以合理的解决。
  3. 要综合考虑系统涉及的各种因素，设计并开发出满足特定要求的软件系统或功能部件。在解决问题时，鼓励采用创造性思维方式解决问题。
  4. 在设计与开发过程中，应选择主流的开发工具及相关资源，以保证系统的先进性、可兼容性和可扩展性。
  5. 在完成功能完善、技术复杂的系统时，允许分组分工，但每个学生应努力完成自己的职责并彼此配合。
  6. 应认真听取专家讲座，了解本专业的前沿知识，并写入生产实习报告。
  7. 在企业参观与交流时，应严格遵守纪律，保证自身及企业生产安全，认真听取企业专家的介绍，仔细查看了解生产车间各要素的功能，并写入生产实习报告。
  8. 验收时，在实验环境下回答指导教师的提问，力求准确、全面、流利。
  9. 撰写依据可靠、方案完整、数据准确、格式规范的实习报告。

**课题要求**

要求:实现眼底图像病灶检测及辅助疾病诊断。

要求:实现MRI影像心脏组织分割与疾病识别。

要求:实现低剂量CT增强同时保证CT成像质量。

以小组为单位，互相协作，分工明确；严格出勤；认真完成生产实习报告，并提交程序。

## 实习内容

* 1. 糖尿病视网膜病变分级
  2. 心室分割与心脏病分类
  3. 低剂量CT图像质量优化

## 实习单位概况

东北大学国家级计算机实验教学示范中心由计算机类专业实验教学中心和计算机公共基础课实验室两部分组成，共有实验教学用房面积4000平米，现有人员包括教授21人，副教授38人，高级实验人员10人，共承担实验课程62门，平均每年承担课时超过60000人时数。中心坚持以“社会实际需求”为导向，以“强化基础，重视能力，面向应用，鼓励创新”为教学理念，以“CDIO工程教育”为体系模型，结合学校和计算机学科实际情况，始终秉承以培养高水平的人才作为最主要的任务，按照课程基础实践、研究型教学、创新型实验、科技竞赛、科研训练、项目实训及创业孵化多个层次，构建贯彻于本科及研究生教学全过程的符合教学规律、培养学生创新意识、提高学生实践和创新能力的“教学做”三位一体的开放创新型实验的规范化教学体系。积极推进示范中心课程体系改革和教学内容建设，切实提高教学质量；以“开放性”为中心，加强工程实践环节，打造一流开放实验教学环境；加强特色专业建设，提高人才培养质量；加强校企联合，产学研合作和实习基地建设。

中心针对计算机专业和非计算机专业学生的差异制定不同的培养计划和目标。面向非计算机专业学生保证计算机基础知识和实践能力的培养，注重介绍计算机在学生主修专业上的应用和扩展，鼓励和帮助学生使用计算机技术对其主修专业进行深度探究和创新实践；面向计算机类专业，注重计算机实践能力的宽度和深度培养，结合理论知识和实际应用提升学生的实践能力，并注重实验扩展和课程间整合提高学生复杂工程性问题解决能力，深入挖掘学生潜力进行扩展性和开放性教学，通过暑期生产实习和夏令营充分引领学生参与课外科技创新活动和研究性学习；利用云技术改造了实验设备和实验室环境，积极探索在新型远程化实验教学系统和开放性实验系统的应用，为学生在实验室外和课余实践学习提供了条件。

在多年的建设和发展过程中，示范中心先后完成了省部改革项目30余项，中心老师和学生共发表学术论文和教改论文100余篇，申请各类专利40余项，指导学生参加各类科技及创新竞赛获奖近400人次，教学成果获得教育部和辽宁省教育厅奖励10余次。示范中心以实验教学改革为抓手，按照3年规划定期组织专家和老师对专业课程设置、教学模式、实践教学案例等方面对实验教学都进行改进和调整，增加了基础课程和专业课程的开放综合性实验项目及复杂工程性实验项目。以实际应用为导向，联合20余家业内知名企业，引入符合教学需求的实验环境和实验课程，学校老师和企业老师共同参与，让学生能实际参与企业工程项目的开发过程，以培养出国家需要、社会需要、企业需要的水平一流的创新型人才。

# 眼底图像病灶检测及辅助疾病诊断

## 任务描述

### 任务背景

数以百万计的人患有糖尿病性视网膜病变，这是老年人失明的主要原因。人们希望在难以进行医疗筛查的农村地区发现和预防这种疾病，同时降低医疗发达地区此类疾病的检测成本。临床上，糖网通常以新生血管为分界，可分为非增值性糖尿病视网膜病变（NPDR）和增值性糖尿病视网膜病变（PDR），其中NDPR又分为轻度、中度、重度三类。

### 相关工作

基于深度学习的图像分类方法为我们的任务提供了显著的帮助。Alex Krizhevsky在2012年ILSVRC提出的CNN模型取得了历史性的突破，效果大幅度超越传统方法，获得了ILSVRC2012冠军，该模型被称作AlexNet。这也是首次将深度学习用于大规模图像分类中。

从AlexNet之后，涌现了一系列CNN模型，不断地在ImageNet上刷新成绩，如图5展示。随着模型变得越来越深以及精妙的结构设计，Top-5的错误率也越来越低，降到了3.5%附近。而在同样的ImageNet数据集上，人眼的辨识错误率大概在5.1%，也就是目前的深度学习模型的识别能力已经超过了人眼。

由于医学图像具有较易于分辨的病理特征供机器学习，因此使用人工神经网络进行疾病的分类诊断可以取得比较理想的效果。

### 任务描述

本任务目标为实现眼底糖网疾病的辅助分类及诊断，即输入一张眼底影像，输出病人对应的疾病结果（无疾病、轻度、中度、重度）。

## 算法总体设计

我们首先对图像数据集进行预处理，主要方式为高斯模糊。

之后选择要使用的模型，经过实验，我们选择了效果最好的EfficientNet。

最后训练模型，进行推理并评估训练效果。

同时，我们使用了迁移学习的思想，即使用预训练模型提高模型的预测效果和收敛速度。

流程图如下所示

模型评估

模型训练

迁移学习

数据处理

## 数据处理

### 数据来源

我们使用了指导教师提供的数据集作为主要训练数据集。此外，我们还使用了Asia Pacific Tele-Ophthalmology Society数据集用作迁移学习。APTOS数据集的实际分布如下图所示。（我们删除了此数据集中的类别4，与指导教师提供的数据集进行同步）。

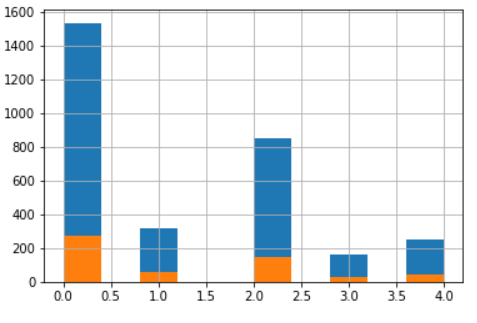


图2.1 APTOS数据分布

### 数据预处理

我们对数据预处理的核心思想为转灰度裁剪后进行高斯模糊。两个数据集均采取了同样的图像预处理方式，使用这种方式处理数据，一是可以减轻光照对图像数据的影响，二是可以裁减掉不提供有用信息的区域，减轻干扰。综上所述，数据预处理的目的是统一化原有数据集，从而提高模型的泛化能力。

## code from https://www.kaggle.com/code/ratthachat/aptos-eye-preprocessing-in-diabetic-retinopathy?scriptVersionId=20340219

# 图像预处理函数

def crop\_image1(img,tol=7):

    # img is image data

    # tol  is tolerance

    mask = img>tol

    return img[np.ix\_(mask.any(1),mask.any(0))]

def crop\_image\_from\_gray(img,tol=7):

    if img.ndim ==2:

        mask = img>tol

        return img[np.ix\_(mask.any(1),mask.any(0))]

    elif img.ndim==3:

        gray\_img = cv2.cvtColor(img, cv2.COLOR\_RGB2GRAY)

        mask = gray\_img>tol

        check\_shape = img[:,:,0][np.ix\_(mask.any(1),mask.any(0))].shape[0]

        if (check\_shape == 0): # image is too dark so that we crop out everything,

            return img # return original image

        else:

            img1=img[:,:,0][np.ix\_(mask.any(1),mask.any(0))]

            img2=img[:,:,1][np.ix\_(mask.any(1),mask.any(0))]

            img3=img[:,:,2][np.ix\_(mask.any(1),mask.any(0))]

    #         print(img1.shape,img2.shape,img3.shape)

            img = np.stack([img1,img2,img3],axis=-1)

    #         print(img.shape)

        return img

### 数据增强

数据增强（Data Augmentation）是一种通过让有限的数据产生更多的等价数据来人工扩展训练数据集的技术。 它是克服训练数据不足的有效手段，目前在深度学习的各个领域中应用广泛。 我们组采用了rand augment[1]进行数据增强，其包括随机裁剪、随机水平翻转等一系列数据增强措施，有效地提高了模型的泛化能力，提高了收敛速度。

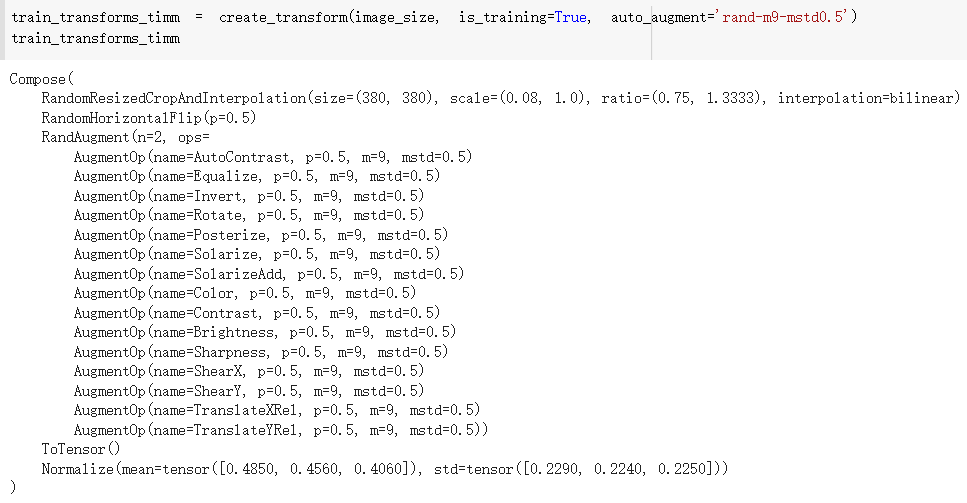


图2.2 RandAugment组成

## 模型选择

图像分类是计算机视觉的核心任务之一，近年来发展十分迅速，有各式各样的网络结构可以供我们选择。我们在模型选择方面进行了较长时间的实验，最终选取了效果最好的EfficientNet来完成我们的任务。

EfficientNet是一种卷积神经网络架构和缩放方法，使用复合系数均匀缩放深度/宽度/分辨率，参数量大大减少。其设计了一个基线网络，并扩展为一系列EfficientNet模型[2]。其探讨了放大传统CNN的深度、宽度、分辨率对最终结果的影响。我们最终选择EfficientNet家族中的EfficientNet\_b4来进行模型训练。

## 模型训练

### 迁移学习

在许多机器学习和数据挖掘算法中，一个主要的假设是训练数据和未来数据必须在相同的特征空间中（即IID）。然而，许多实际应用中，这种假设不可能成立。在这种情况下，如果成功进行知识迁移，就可以避免昂贵的数据标注工作，从而大大提高学习的性能[3]。

迁移学习的核心是找到已有知识和新知识之间的相似性。我们使用迁移学习的思想来训练我们的模型，这将在2.5.2中进行详细阐述。

我的主要工作是进行模型的训练与修正，在实验中调整合适的参数，迁移学习的有效性是我在实验中经过思考和查证得到的，是我们工作创新性的主要体现。

### 模型预训练

如上一小节所述，使用迁移学习之前，需要进行预训练。我们进行预训练的步骤如下：

1. 对预训练数据集图片不进行预处理步骤，仅使用数据增强来进行训练，得到预训练模型。
2. 对正式训练数据集图片进行预处理，导入预训练模型进行训练，得到最终模型。

我们的实验结果表明，使用迁移学习可以提高第二次训练模型的收敛速度，同时可以一定程度上提高模型的精度。

根据论文作者的建议，预训练设置image\_size为380，同时设置batch\_size为16，learning\_rate为5e-5。使用交叉熵损失函数和Adam优化器进行30轮训练。训练核心部分代码如下（省略部分变量）：

criterion = nn.CrossEntropyLoss()

optimizer = optim.Adam(model.parameters(), lr=lr, weight\_decay=1e-5)

scheduler = StepLR(optimizer, step\_size=3, gamma=0.5)

save\_path = 'effi\_b4\_380x380.pth'

# Train the model

total\_step = len(train\_dataloader)

for epoch in range(epochs):

    # train

    model.train()

    for i, (images, labels) in enumerate(tqdm(train\_dataloader)):

        images = images.to(device)

        labels = labels.to(device)

        # Forward pass

        outputs = model(images)

        loss = criterion(outputs, labels)

        train\_loss += loss.item()

        \_, predicted = torch.max(outputs.data, 1)

        total += labels.size(0)

        correct += (predicted == labels).sum().item()

        # Backward and optimize

        optimizer.zero\_grad()

        loss.backward()

        optimizer.step()

    # validate the model

    model.eval()  # eval mode (batchnorm uses moving mean/variance instead of mini-batch mean/variance)

    val\_loss = 0.0

    with torch.no\_grad():

        correct = 0

        total = 0

        for images, labels in valid\_dataloader:

            images = images.to(device)

            labels = labels.to(device)

            outputs = model(images)

            loss = criterion(outputs, labels)

            val\_loss += loss.item()

            \_, predicted = torch.max(outputs.data, 1)

            total += labels.size(0)

            correct += (predicted == labels).sum().item()

    if (val\_loss / len(valid\_dataloader)) < best\_loss:

      best\_loss = val\_loss / len(valid\_dataloader)

    if (100 \* correct / total) > best\_acc:

      best\_acc = 100 \* correct / total

      print('Save best model,loss: {:.4f}  acc: {}'.format(best\_loss, best\_acc))

      torch.save(model.state\_dict(), save\_path)

### 模型训练

预训练的目的为取得原始预训练数据集图像的信息作为先验知识，将其运用于我们数据集的正式训练中，一定程度上相当于扩充了数据集。模型的正式训练需要首先载入预训练权重文件，使用与预训练相同的模型进行100轮训练。不同的是，正式训练使用的数据集经过了2.3.2小节叙述的预处理模块，同时使用动态学习率模块，在loss在一段时间内不下降时，调整学习率的值。

由于对比实验的重要性，我们设置了三个大的实验组，分别为：(1) 使用APTOS预训练，指导教师数据集正式训练。 (2) 使用APTOS预训练，使用APTOS正式训练。 (3) 使用指导教师数据集预训练，使用指导教师数据集正式训练。实验结果的分析见2.6节。

## 实验分析

### 不同数据集的迁移结果讨论

我在最开始进行实验时，仅仅使用的是APTOS数据集进行训练。由于经验和知识储备不够充足，我们犯下了一个比较严重的错误——想当然地认为APTOS数据集上训练的模型，直接用于预测目标数据集同样会取得比较理想的效果。然而在实际测试时，在APTOS数据集上训练的模型，在我们目标数据集上预测的表现奇差无比，这迫使我们更改实验方案，重新进行训练。经过这次错误，我们深刻认识到了数据分布和特征对模型影响程度之大，同时希望能找到一种办法，一定程度上减轻数据特征空间对模型的影响，即提高模型的泛化能力。

### 训练结果分析

我们首先在APTOS数据集上进行训练，依然是使用迁移学习的思想，将未经预处理的图像作为先验知识。训练结果比较理想，预训练模型最终的验证集acc稳定在86.4%，正式训练模型验证集acc稳定在87.6%，提高了一个百分点。同时不经预训练的最终acc大约为85%，迁移学习对最终效果有显著提高。

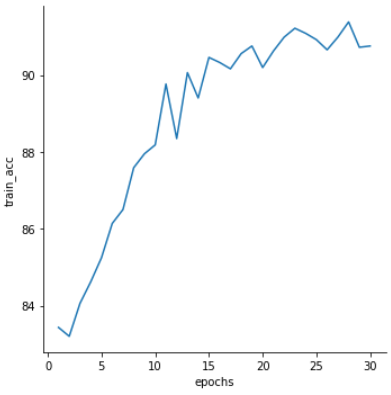
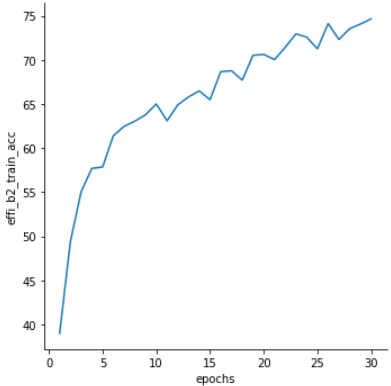
 

图2.3 数据集一训练集acc 图2.4 数据集二训练集acc

之后，我们在指导教师提供的数据集上使用我们的训练方式进行训练。预训练验证集acc为75%，正式训练验证集acc为79%。部分训练结果图见图2.4。

### 问题探讨

1. 模型在两个数据集之间的泛化能力较弱。这是我们本次实验中最重要的一个问题，在2.6.1中进行了阐述。我们分析的原因之一，可能是APTOS采集的是亚洲人眼底数据，而指导教师提供的数据集眼底数据可能与APTOS在种族方面存在一定的差异，在模型训练过程中，这种微小的差异被无限放大。
2. 验证集准确率抖动现象。如图2.5、2.6所示，训练过程中验证集loss和acc的抖动较为明显，我们尝试了多种方法，这种现象依然存在。我们推测其原因可能是：

* 数据集图片数量小且分布不均。
* 由于数据量的不足，为了提高模型的泛化能力，广泛使用了数据增强和正则化，导致分布差距进一步加大。
* Dropout的影响。
* 预训练时损失稳定下降，正式训练时接近收敛，持续在最优解附近震荡。

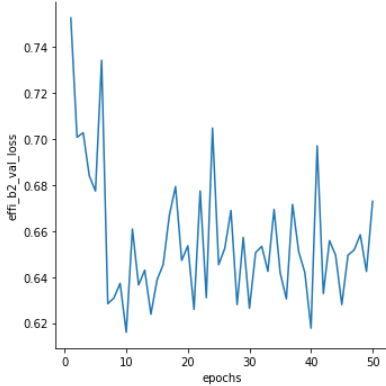
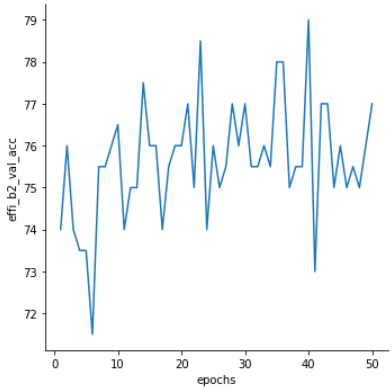
 

图2.5 正式训练验证集loss 图2.6 正式训练验证集acc

# 

# 心室分割与疾病分类

## 任务描述

### 任务背景

心脏疾病是影响人类健康的最主要疾病之一，每年无数人因心脏病而去世。核磁共振检查（MRI）对心脏疾病的检测与诊断起到至关重要的作用。其中，对心脏不同组织部位的分割对于区分扩张性心肌病（DCM）、肥厚性心肌病（HCM）和心脏健康的人群具有重要意义。

### 任务描述

本任务目标为对心脏的心房、心室等组织进行语义分割，同时对疾病进行分类。

## 算法总体设计

我们首先对图像数据集进行预处理，主要方式resize和0像素填充。

之后选择要使用的模型。经过实验，针对分割问题，我们选择了效果最好的UNet++。针对分类问题，我们选择使用ResNet。

最后训练模型，进行推理并评估训练效果。

流程图如下所示

模型评估

训练分类模型

训练分割模型

数据处理

## 数据处理

### 数据来源

我们使用了指导教师提供的数据集作为训练数据集。其中共包含400余份数据，每份数据由心脏MRI和分割标签构成，如图3.1和3.2。数据集按照疾病病人排列，并对疾病类型进行了划分。每位病人的图片数量在10张左右。标签包含0,85,170,255四种灰度值，将心脏分割为不同的区域。

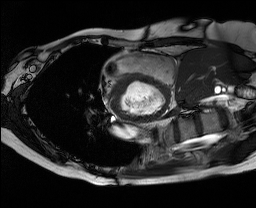
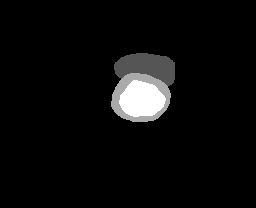
 

图3.1 心脏MRI 图3.2 标签

### 数据预处理

我们主要对图像进行0像素填充来处理原图片。由于原数据集图像尺寸不统一，无法组成batch输入网络；同时考虑到用背景色0填充不会影响分割结果，因此使用这种方式进行处理。部分数据预处理代码如下：

    def preprocess(self, pil\_img, pil\_mask):

        img\_ndarray = np.asarray(pil\_img)

        mask\_ndarray = np.asarray(pil\_mask)

        h, w = img\_ndarray.shape[0:2]

        h1, w1 = mask\_ndarray.shape[0:2]

        top = (self.img\_size - h) // 2

        left = (self.img\_size - w) // 2

        bottom = self.img\_size - h - top

        right = self.img\_size - w - left

        padding = [top, left, bottom, right]

        pixel = 0 if len(img\_ndarray.shape) == 2 else (0, 0, 0)

        image\_padding = cv2.copyMakeBorder(img\_ndarray, top, bottom, left, right, cv2.BORDER\_CONSTANT, value=pixel)

        mask\_padding = cv2.copyMakeBorder(mask\_ndarray, top, bottom, left, right, cv2.BORDER\_CONSTANT, value=pixel)

        image = np.expand\_dims(image\_padding, axis=2)

        mask = np.expand\_dims(mask\_padding, axis=2)

        mask = mask\_to\_onehot(mask, self.palette)

        image = image.transpose([2, 0, 1])

        mask = mask.transpose([2, 0, 1])

        image = image / 255

        return image, mask, padding

### 数据增强

数据增强（Data Augmentation）是一种通过让有限的数据产生更多的等价数据来人工扩展训练数据集的技术。 它是克服训练数据不足的有效手段，目前在深度学习的各个领域中应用广泛。针对此项任务，我们对语义分割任务图像数据不做任何数据增强处理。针对分类任务，我们对输入图像进行随机裁剪和水平翻转，以期提高模型的泛化能力。

## 模型选择

语义分割是计算机视觉的核心任务之一，是将标签或类别与图片的每个像素关联的一种深度学习算法。 它用来识别构成可区分类别的像素集合。其中基于深度学习的语义分割在近年来大放异彩，同时效果也十分理想。UNet是语义分割领域的经典之作，最初也是被提出用于医学图像的分割。UNet以encoder-decoder框架为基础，通过下采样和上采样，得到最终的分割结果[4]。模型结构如图3.3。

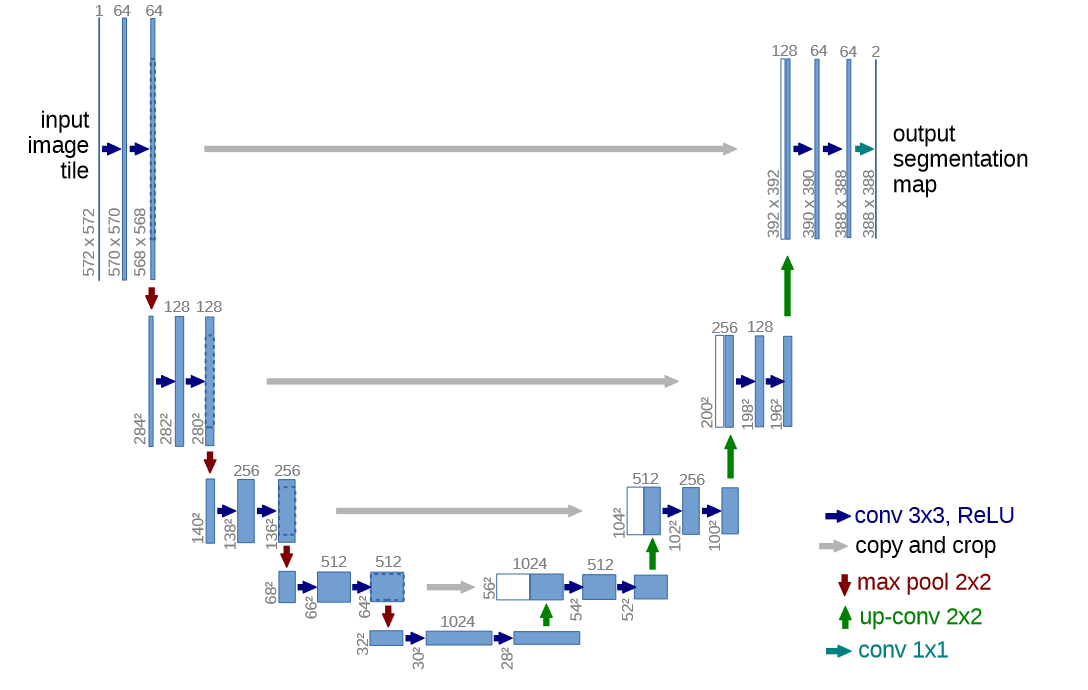


图3.3 UNet网络结构

UNet++是对UNet的一种改进，其综合了长连接和短连接的方案，可以学习不同的深度特征[5]。我们主要使用UNet++模型来完成本任务。经过试验验证，encoder部分，我们选择使用resnet18来完成。其模型结构如图3.4。

我们使用经典网络结构resnet进行图像分类。经过我们的实验验证，浅层resnet18即可以在我们的数据集上获得很好的效果。因此，我们使用resnet18，以groundtruth作为输入训练图像分类网络。并用分割结果推理，得到疾病类型。其残差结构如图3.5。

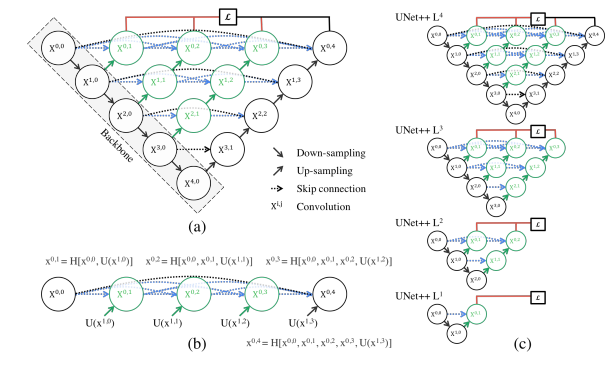


图3.4 UNet++网络结构

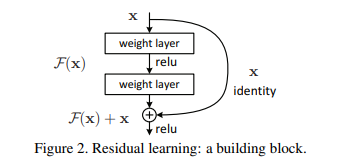


图3.5 ResNet残差模块[6]

## 模型训练

### 语义分割

我们语义分割任务选择使用segmentation\_models\_pytorch（smp）库中的UNet++模型架构进行语义分割任务的训练。根据实验结果与训练环境，设置image\_size为256，batch\_size为16，训练轮数为100，初始学习率为5e-4，在验证集损失不下降时进行衰减。

损失函数使用交叉熵与dice损失的加权平均。Dice系数是一种集合相似度度量函数，通常用于计算两个样本的相似度，取值为[0,1]。其计算公式为：

dice\_loss的值为1-dice。相关研究表明，在类别不均衡时，使用dice系数更加合适。然而，仅仅使用dice系数作为损失函数的一种表示可能是不恰当的，如dice系数可能会忽略小样本的影响。因此我们引入交叉熵损失，这样可以同时考量全局与微观的影响、解决前景背景不均衡的问题、使网络预测正确的学习方向。我们使用目前比较广泛的dice系数和损失实现，代码如下。

class SoftDiceLoss(\_Loss):

    def \_\_init\_\_(self, num\_classes):

        super(SoftDiceLoss, self).\_\_init\_\_()

        self.num\_classes = num\_classes

    def forward(self, y\_pred, y\_true):

        class\_dice = []

        # 从1开始排除背景，前提是颜色表palette中背景放在第一个位置 [[0], ..., ...]

        for i in range(1, self.num\_classes):

            class\_dice.append(diceCoeffv2(y\_pred[:, i:i + 1, :], y\_true[:, i:i + 1, :]))

        mean\_dice = sum(class\_dice) / len(class\_dice)

        return 1 - mean\_dice

class BCE\_Dice\_Loss(\_Loss):

    def \_\_init\_\_(self, num\_classes, smooth=0, weight=[1.0, 1.0]):

        super(BCE\_Dice\_Loss, self).\_\_init\_\_()

        self.bce\_loss = nn.BCEWithLogitsLoss()

        # self.bce\_loss = nn.BCELoss()

        self.dice\_loss = SoftDiceLoss(num\_classes=num\_classes)

        self.weight = weight

        self.smooth = smooth

        self.num\_classes = num\_classes

    def forward(self, inputs, targets):

        return self.weight[0] \* self.bce\_loss(inputs, targets \* (1 - self.smooth) + self.smooth / self.num\_classes) + self.weight[1] \* self.dice\_loss(inputs, targets)

我们使用resnet18网络结构作为UNet++的encoder部分，并随机初始化参数，用以提取图像特征。由于训练代码过长，且与2.5节有一定的相似之处，在此不做额外展示，我们将损失函数替换为上述BCE\_Dice损失，并在每一轮输出模型的Dice系数作为评价指标，从而筛选最优模型。实验结果分析见3.6节。

### 疾病图像分类

我们疾病图像分类选择使用resnet18图像分类网络实现，其中输入为原始图像的分割结果。根据训练需要，设置batch\_size为2（设备显存不足），image\_size为512，初始学习率为1e-5训练300轮。损失函数使用交叉熵，优化器使用权重衰减为1e-5的Adam优化器。最终在我们划分的验证集上的acc可以达到85%。

## 实验分析

### 训练结果分析

根据3.5.1节的训练参数训练UNet++语义分割模型，得到最终验证集的损失函数和dice系数分别如图3.6及3.7。通过训练结果曲线可以得知，模型在30左右即收敛至最优，收敛速度较为理想，同时dice系数可以达到0.87。模型推理时，输入心脏MRI，输出分割后的结果。

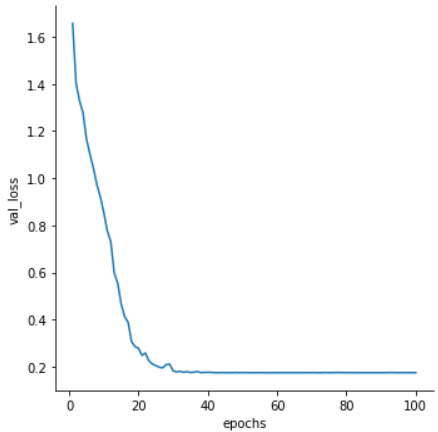
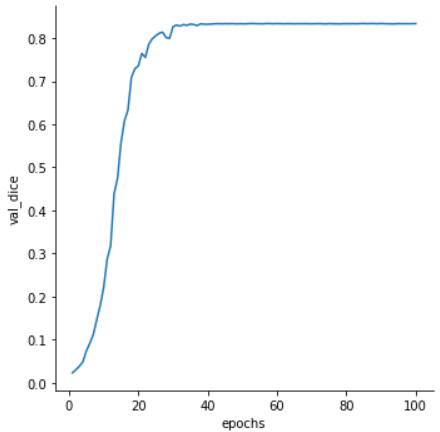
 

图3.6 验证集loss 图3.7 训练集dice系数

根据3.5.2节的训练参数训练ResNet18图像分类模型。得到最终验证集精度为0.85。模型推理时，输入图像分割网络得到的结果，输出疾病分类。

### 问题探讨

1. 图像分类网络训练时acc抖动较大。我们推测可能为数据集标签的分布比较广泛，网络难以学习到更加合适的参数，因此引起抖动现象。
2. 语义分割模型不同像素类别训练得到的dice系数有差异。我们通过实验发现，在标签中颜色越浅像素类的dice系数越低，即分割效果差。我们通过观察原始数据发现，像素值小的心脏组织在影像学上特征并不明显，因此网络训练时，难以学习合适的分割参数。为了消除这类问题，我们尝试调整网络结构，并更换其他语义分割网络，并未取得理想的效果。所以我们得出了一个可能的推测——由于数据对模型起决定性作用，因此单独更改和调整模型对这类问题起到的作用较为有限，需要从数据入手解决这方面的问题。

# 基于CycleGAN的低剂量CT质量优化

## 任务描述

### 任务背景

CT(Computed Tomography)，即计算机断层成像，其作为医学影像领域最重要的成像技术之一，能够实现快速的高空间分辨率、高密度分辨率三维成像，被广泛应用于神经系统、心血管系统、软组织器官以及骨骼等疾病的临床诊断。CT利用X射线管环绕人体发射出X射线，X射线被探测器接收后，经过数据校正及图像重建算法即可重建出人体组织对X射线的衰减系数分布图，即CT断层图像。

在这一过程中，由于需要利用X射线照射人体，因此CT扫描过程会对人体造成一定剂量的电离辐射，对患者的健康造成潜在的危害。

为了尽可能的降低CT扫描对人体的伤害，可以通过降低CT扫描的扫描电流或电压来达到降低辐射剂量的目的，但降低扫描电流或电压，会直接导致图像噪声的增加，降低CT图像质量。因此，如何实现低放射剂量的同时保证CT成像质量一直是CT成像领域研究的重点。

### 研究现状

近些年，深度学习技术被广泛应用于图像的优化、检测以及分类等领域，并成为医学图像优化领域中的研究热点。在一些公开数据集中，相比于传统方法，基于深度学习的CT图像优化方法表现出了显著的优势，但深度学习技术在CT临床图像优化中的应用仍然面临一些困难：首先，仿真数据较难覆盖临床设备的各种扫描参数及模拟设备间差异，因此基于仿真数据训练的模型难以在临床中应用；同时，也难以采集到结构信息完全匹配的CT低剂量和高剂量图像用于有监督训练。因此，亟需创新性的方法的提出，以解决现有的低剂量CT增强方面所面临的困难。

### 任务描述

本任务目标为对低剂量CT进行去噪，模拟高剂量CT影像。

## 算法总体设计

我们首先对图像数据集进行分析预处理。

之后选择要使用的模型。经过实验，我们选择使用CycleGAN进行训练。

最后训练模型，进行推理并评估训练效果。

## 数据处理

### 数据来源

我们使用了指导教师提供的数据集作为训练数据集。数据集为病人CT影像，分为raw文件和mhd文件。Raw文件保存的是纯像素信息。常常是一个病人的所有dicom文件中的图像提取出来放在一个raw文件里。即一个病人对应一个raw文件，其中存储的是该病人的图像信息。（也可以理解为将该病人不同的dicom切片图像都叠到一起，形成了一个三维图像）。dicom文件除了包含切片图像外，还包含其他的一些信息。在文件格式转换后，图像信息被raw文件提取，非图像信息则存储在mhd头文件中。即mhd头文件是存储关于一个病人的所有dicom文件中的非图像信息。处理前的图像数据如图3.1所示。

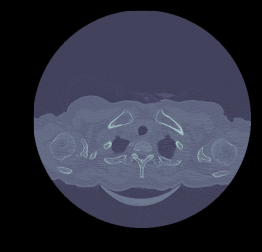


图4.1 原始CT影像

### 数据预处理

我们主要对图像进行归一化并调整窗位和窗宽。使用 SimpleITK 库进行mhd和raw文件的读取，并使用cv2.normalize归一化，将指定图片的值放缩到 0-255 之间。Image.fromarray实现array到image的转换，转换完成后进行保存。预处理之后的图像如图3.2和3.3所示。

图4.2 预处理后低剂量CT影像 图4.3 预处理后高剂量CT影像

## 模型选择

生成对抗网络（GAN）由Ian Goodfellow于2014年提出，是一种非监督学习方法，由两个神经网络（生成器和判别器）相互博弈的方式进行学习[7]。经典的生成对抗网络如图3.4所示。基于生成对抗网络，各种各样的GAN不断涌现，如WGAN，CycleGAN等。在CT影像去噪方面，GAN家族取得了较为良好的应用，Qingsong Yang等人使用WGAN在临床CT影像去噪方面取得了较好的效果[8]。

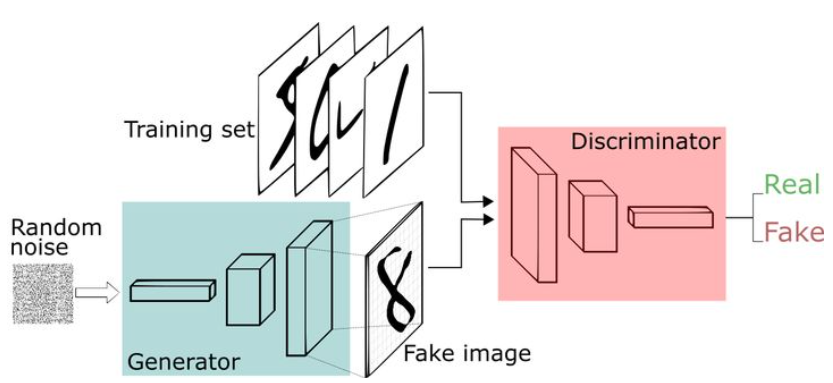


图4.4 GAN结构

经过实验对比，我们选择更适合我们数据的CycleGAN训练模型。循环生成对抗网络（CycleGAN）的一个最大特点是无监督，也就是不要求训练数据是成对的，只需要提供不同域（domain）的图像即可完成图像数据之间的映射（风格迁移）[9]。因此十分适合我们本次的任务。CycleGAN的结构如图3.5所示。

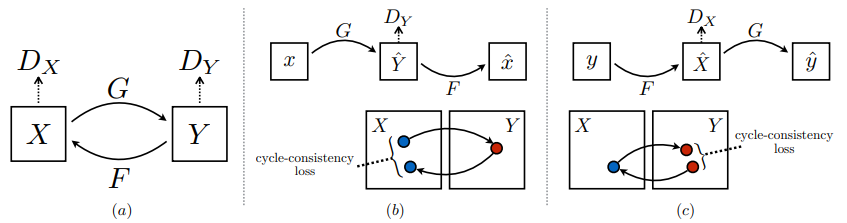


图4.5 CycleGAN[9]

## 模型训练

### 模型训练

GAN模型训练时的batch\_size通常设置为1，因此我们训练时也将batch\_size设置为1。我们将输入图片的大小设置为512，使用随机裁剪的方式调整图片大小为256后输入网络进行训练。生成器使用ResNet网络，以2e-4的固定学习率进行100轮训练。实验表明，模型在25轮左右收敛至最优，后续轮数由于过拟合会有较为严重的伪影现象。因此，我们选择第25轮的模型为最终结果。部分训练代码如下：

if \_\_name\_\_ == '\_\_main\_\_':

    opt = TrainOptions().parse()

    dataset = create\_dataset(opt)

    dataset\_size = len(dataset)

    print('The number of training images = %d' % dataset\_size)

    model = create\_model(opt)

    model.setup(opt)

    visualizer = Visualizer(opt)

    total\_iters = 0

    for epoch in range(opt.epoch\_count, opt.n\_epochs + opt.n\_epochs\_decay + 1):

        epoch\_start\_time = time.time()

        iter\_data\_time = time.time()

        epoch\_iter = 0

        visualizer.reset()

        model.update\_learning\_rate()

        for i, data in enumerate(dataset):

            iter\_start\_time = time.time()

            if total\_iters % opt.print\_freq == 0:

                t\_data = iter\_start\_time - iter\_data\_time

            total\_iters += opt.batch\_size

            epoch\_iter += opt.batch\_size

            model.set\_input(data)

            model.optimize\_parameters()

            if total\_iters % opt.display\_freq == 0:

                save\_result = total\_iters % opt.update\_html\_freq == 0

                model.compute\_visuals()

                visualizer.display\_current\_results(model.get\_current\_visuals(), epoch, save\_result)

            if total\_iters % opt.print\_freq == 0:

                losses = model.get\_current\_losses()

                t\_comp = (time.time() - iter\_start\_time) / opt.batch\_size

                visualizer.print\_current\_losses(epoch, epoch\_iter, losses, t\_comp, t\_data)

                if opt.display\_id > 0:

                    visualizer.plot\_current\_losses(epoch, float(epoch\_iter) / dataset\_size, losses)

            if total\_iters % opt.save\_latest\_freq == 0:

                print('saving the latest model (epoch %d, total\_iters %d)' % (epoch, total\_iters))

                save\_suffix = 'iter\_%d' % total\_iters if opt.save\_by\_iter else 'latest'

                model.save\_networks(save\_suffix)

            iter\_data\_time = time.time()

        if epoch % opt.save\_epoch\_freq == 0:

            print('saving the model at the end of epoch %d, iters %d' % (epoch, total\_iters))

            model.save\_networks('latest')

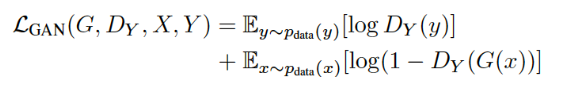
            model.save\_networks(epoch)

        print('End of epoch %d / %d \t Time Taken: %d sec' % (epoch, opt.n\_epochs + opt.n\_epochs\_decay, time.time() - epoch\_start\_time))

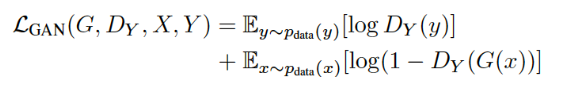
### 损失函数

CycleGAN的损失较为复杂，因此在此做出一些说明。

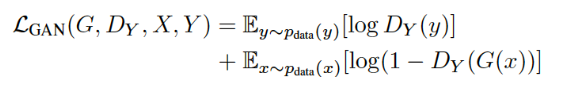
Adversarial loss：尽可能让生成器生成的数据接近真实的数据分布。



Cycle consistency loss：用于让两个生成器生成的样本之间不要相互矛盾。



总体损失：生成器G尽可能实现X到Y的迁移，生成器F尽可能实现Y到X的迁移。同时，希望两生成器可以实现互逆，即通过迭代后可以互相表示。



Idt loss：防止输入和输出之间的color compostion过多。

### 训练结果分析

根据4.5.1节进行训练后，选择第25轮的训练结果作为最终模型。训练轮数过多时，会有严重的伪影现象，造成图片失真；而训练轮数过少，会导致去噪效果不佳。训练完成后会得到两个生成器模型，我们选择低剂量向高剂量迁移的生成器模型进行测试和推理。经实验验证，模型在470张图片的测试集上运行时间为1分41秒，运行速度较快。

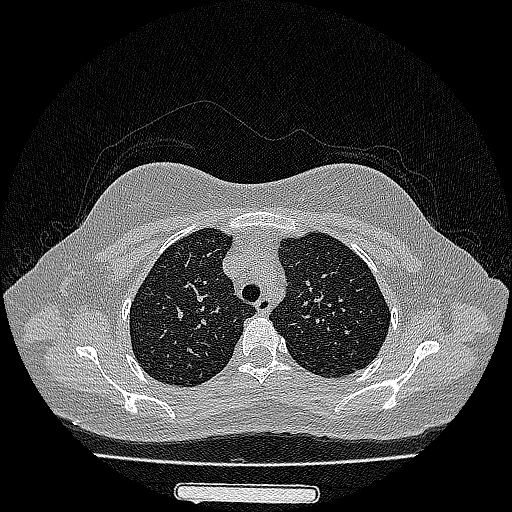
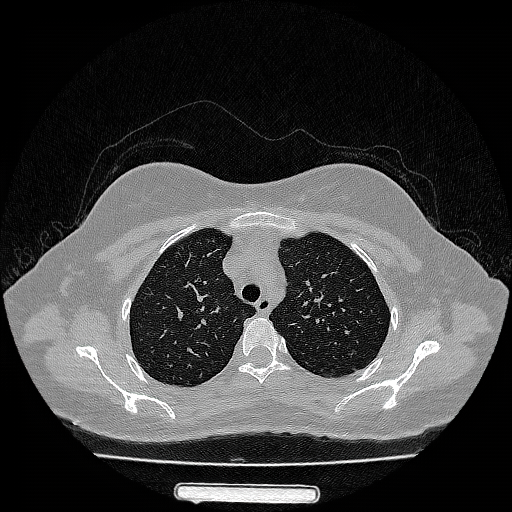
 

图4.6 测试集低剂量CT 图4.7 生成器输出高剂量CT

根据4.5.2节所述，训练过程中的相关损失函数由图4.8-4.11所示。可以发现，训练过程中cycle\_A和idt\_A的值在不断下降。由于GAN的特殊性，完全由损失函数判断模型效果是不可取的。因此我们在每一轮的训练中间，都抽取出了一些图片观察训练效果，并用此作为最终模型选择的依据。

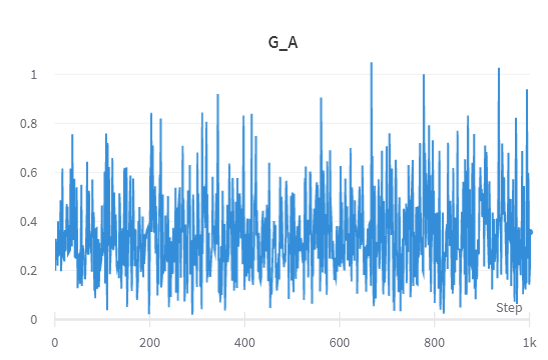
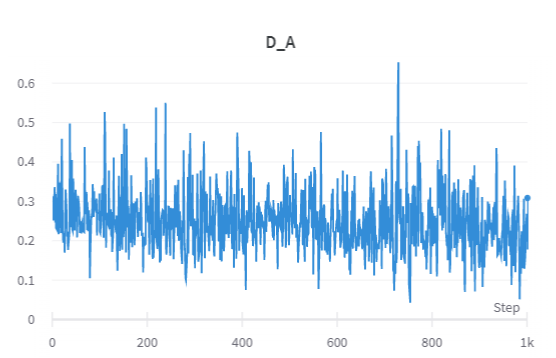
 

图4.8 G\_A 图4.9 D\_A

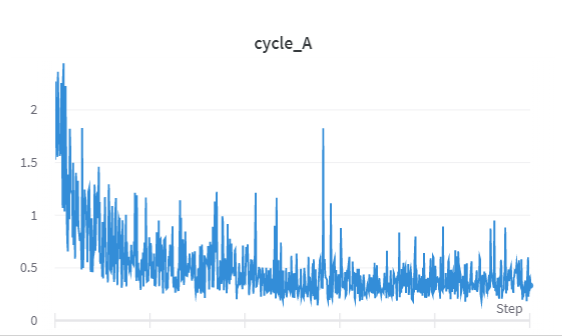
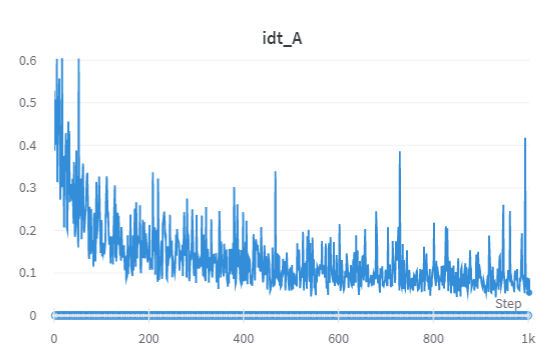
 

图4.10 cycle\_A 图4.11 idt\_A

### 问题探讨

由于GAN的训练过程较为复杂，因此我们在训练过程中遇到了一些问题，在此进行一些探讨。

1.图片设置。我们经过实验，最终选取输入图像512，经过随机裁剪后，以256的大小输入网络进行训练。这样做的原因是，以高分辨率输入图片会使图片效果更好；同时经过裁剪，会突出图像的细节特征，也会提高模型的最终效果。在推理时，我们仅需调整图片的输入输出尺寸即可。

2.生成器模型。生成器选择不同的网络结构会对风格迁移效果产生影响，经过测试，我们选择有9个block的ResNet进行训练。

3.伪影现象。如上文所述，训练轮数过多时会有伪影现象产生，如图3.12。产生这种现象的原因之一是过拟合，因此我们需要在训练过程中严格监控伪影现象的产生，防止训练过度。

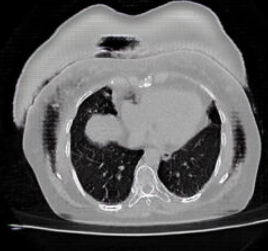


图3.12 伪影现象

# 实习总结

## 实习内容的复杂性评价

本次实习的工程复杂性评价如表4.1所示：

表4.1 工程复杂性评价表

|  |  |  |
| --- | --- | --- |
|  | **复杂工程问题特征** | **问题描述及解决方案（方法）** |
| 1 | 必须运用深入的工程原理，经过分析才可能得到解决 | 不同数据域训练的模型很难进行迁移  考虑进行模型预训练，可以使问题得到一定程度的改善 |
| 2 | 涉及多方面的技术、工程和其它因素，并可能相互有一定冲突 | 训练速度过慢，需要使用一些第三方库进行训练加速 |
| 3 | 需要通过建立合适的抽象模型才能解决，在建模过程中需要体现出创造性 | 针对不同数据如何选取合适的模型和参数，需要经过多次实验建立系统方法 |
| 4 | 不是仅靠常用方法就可以完全解决的 | 语义分割多分类任务输入时需要调整编码格式 |
| 5 | 问题中涉及的因素可能没有完全包含在专业工程实践的标准和规范中 | 我们本次实训仅在模型训练过程下了很大功夫，部署方面等工程问题并没有仔细研究 |
| 6 | 问题相关各方利益不完全一致 | 无 |
| 7 | 具有较高的综合性，包含多个相互关联的子问题 | 核心训练代码需要考虑数据问题、硬件问题、如何可视化问题，还需考虑下一步如何调整，具有较高的综合性 |

## 实习体会、收获与建议

通过本次实习，我得以将我大学前三年所学到的计算机基础知识和人工智能基础知识得以综合运用，来解决一些实际问题，学习算法设计与实验的具体过程与细节，收获颇多，在此对这段时间学习和工作的收获做出一点总结。

实践是检验真理的唯一标准，“知行合一”的校训是每一个东北大学学子信奉的真理。只有经过实践，所学的理论才能真正变成我所掌握的知识。不经过实践的学习是不完整的，也是学不到真正的本领的。作为大学生，想要不断成长，就需要勤于实践，将理论与实践结合起来，提高自己的综合素质。

经过这一段时间的学习与工作，我得以学习到了很多在课堂上学习不到的东西，学习掌握了多项科研技能和工程技能，丰富了自身经历，完善了知识体系，提高了多种技术。经过这样的实习训练，才能在不同的任务中不断提高自己分析、解决问题的能力。编写、修改、优化代码的过程，是对自己所学知识的检验，更是对自己工作技能和心智的一种历练。

纸上得来终觉浅，绝知此事要躬行。亲力亲为，亲手实践，是提高自身能力的唯一途径。理论课的学习往往过于表面，只有深入到实践当中去，才能理解那些经典理论的伟大现实意义。同时由于工程的复杂性，实践的过程中会遇到大大小小的各种问题，这都需要运用工程技能去解决，仅仅了解理论，是无法解决问题的，只能当一个不懂技术、只知高谈阔论的“做题家”。工科生不能允许自己没有解决实际问题的能力，这需要在实践中不断培养。

细节决定成败，代码也是一样。工程中会遇到种种细节问题，经常一个微小的错误会导致整个系统的崩溃。算法也是一样，任何一个细节都会对最终的结果有显著影响。我们在理论课上，无法面面俱到地学习各种各样的细节问题，没有时间，也没有必要。我们在实践中，会逐渐理解、学会解决细节问题，为将来走向工作打下良好的基础。

初次进行生产实习，经验不足，在实习过程中，深感储备不够、能力不够。工作经常因为一些现在看起来很小的问题而停滞。在三位指导教师的帮助下，在小组成员的共同努力下，我们可以解决一个又一个的问题，完成我们的最终目标。身为组长，需要统筹规划小组成员的分工，并且完成较为困难的工作，这对我来说是一个不小的挑战。坚持就是胜利，在我们的不懈努力下，最终还是圆满的完成了实践。

感谢学校为我们提供了如此宝贵的实习机会，我们可以真正将我们所学运用于实际工作中去，而不仅仅是停于表面。感谢三位教授对我们的鼎力支持，没有老师们的辛勤工作和敦敦教导，我们肯定无法将工作顺利进行下去。感谢队友们的付出，没有与你们的合作，凭借一个人的力量完成如此庞大且复杂的工程是很困难的。

# 参考文献

1. Cubuk, Ekin D., et al. "Randaugment: Practical automated data augmentation with a reduced search space." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2020.
2. Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.
3. Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." *IEEE Transactions on knowledge and data engineering* 22.10 (2009): 1345-1359.
4. Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*. Springer, Cham, 2015.
5. Zhou, Zongwei, et al. "Unet++: A nested u-net architecture for medical image segmentation." *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, Cham, 2018. 3-11.

[6] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[7] Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems* 27 (2014).

[8] Yang, Qingsong, et al. "Low-dose CT image denoising using a generative adversarial network with Wasserstein distance and perceptual loss." *IEEE transactions on medical imaging* 37.6 (2018): 1348-1357.

[9] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." *Proceedings of the IEEE international conference on computer vision*. 2017

# 附录 实习日记

|  |  |
| --- | --- |
| 5月11日 | 上午正式开题，在老师的讲述下，我们对于本次实训有了初步的认识，然后进行了自由组队。 |
| 5月14日 | 上午曹鹏老师讲解本次项目题目，带领大家对医学影像处理领域方面的知识进行深入理解；下午成员查找相关资料，选择模型。 |
| 5月17日 | 选择合适模型，并编写相关代码，完成数据处理相关内容。 |
| 5月18日 | 测试各种模型，寻找不同数据集进行训练测试 |
| 5月21日 | 确定对应模型，优化模型参数 |
| 5月25日 | 调整模型参数，完成界面布置，实现项目可视化。 |
| 5月27日 | 上午对项目进行了全面检查和完善，并提交给老师进行程序验收。 |
| 5月28日 | 上午冯朝路老师讲解本次项目题目，带领大家对医学影像处理领域方面的知识进行深入理解；下午成员查找相关资料，选择模型。 |
| 5月30日 | 选择合适模型，并编写相关代码，完成数据处理相关内容。 |
| 6月1日 | 测试各种模型，使用老师提供数据集进行测试 |
| 6月3日 | 确定对应模型，优化模型参数 |
| 6月4日 | 调整模型参数，完成界面布置，实现项目可视化。 |
| 6月6日 | 上午对项目进行了全面检查和完善，并提交给老师进行程序验收。 |
| 6月7日 | 上午栗伟老师讲解本次项目题目，带领大家对医学影像处理领域方面的知识进行深入理解；下午成员查找相关资料，选择模型。 |
| 6月8日 | 选择合适模型，并编写相关代码，完成数据处理相关内容。 |
| 6月9日 | 测试各种模型，寻找不同数据集进行训练测试 |
| 6月11日 | 确定对应模型，优化模型参数 |
| 6月13日 | 调整模型参数，完成界面布置，实现项目可视化。 |
| 6月15日 | 上午对项目进行了全面检查和完善，并提交给老师进行程序验收。 |

|  |  |  |
| --- | --- | --- |
| 指导教师姓名： | | |
| 是否有专题调研 | |  |
| 专题题目 |  | |
| 实习单位意见：  签字: 年 月 日 | | |
| 指导教师意见:  成绩： 签字: 年 月 日 | | |
| 备注: | | |