福州大学

本科生毕业设计（论文）

题　　目：基于深度学习的舌象图像合成模型研究

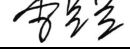
姓　　名：王明石

学　　号：112100403

学　　院：物理与信息工程学院

专　　业：电子信息工程

年　　级：2021 级

校内指导教师：  （签名）

校外指导教师：　　（签名）

2025 年 5 月 27 日

基于深度学习的舌象图像合成模型研究

摘　　要

近年来深度学习模型对医学图像的质量要求和依赖度越来越高，而这种需求对医学影像而言存在着诸多困难与限制：采集的医学图像存在隐私、伦理等原因致使很难公开获得高质量数据；正负样本数据不平衡；由于光线条件等因素引起的域间噪声让现有数据无法与真实数据保持一致。对舌象图像而言，“缺乏数据—数据分布不均衡—域间复杂”更为突出。

为解决以上问题，本文分别采用 DCGAN 和 DDIM 两种模型合成了大量高质量的舌象图像，完成了扩充舌象图像数据集和提升分类模型效果的任务。鉴于舌象图像区别于自然图像的独特纹理和色彩特征，本文对其损失函数进行了创新改进，在 DDIM 原版损失函数的基础上引入了颜色直方约束、结构相似性损失与感知损失等损失函数，提出了 Tongue-DDIM 舌像图像合成框架。

为验证 Tongue-DDIM 在舌像图像合成任务上对 DCGAN 的优势，本文以医院获取的舌象图像数据集分别在两种模型上进行训练和推理。为了对比 DCGAN 和 Tongue-DDIM 的合成效果，本文引入 FID 和 IS 两个技术指标，证明了 Tongue-DDIM 在细节表达程度和纹理的真实性方面较传统 DCGAN 模型有明显提升，且优于没有改进损失函数的 DDIM。

为验证舌象图像合成方法对分类任务数据集扩充的有效性，利用两种模型新合成的舌象图像数据，分别构建了两种扩充数据集。本文将扩充数据集应用于舌象-结直肠癌分类任务。实验结果证明扩充数据集大幅提高了分类准确率和其他性能指标。本研究验证了扩散合成模型可以实现医学舌象图像数据集扩充任务，构建了针对舌象图像特征优化的扩散合成模型。本研究构建的舌象图像扩充数据集，可为针对结直肠癌（Colorectal Cancer，CRC）的非入侵性早期筛查及临床诊断提供高质量数据支持，并为基于人工智能的 CRC 诊治研究探索出一条技术之路。

关键词：深度学习，医学图像，扩散模型，Tongue-DDIM，DCGAN

Research on Deep Learning-Based Tongue Images Synthesis Models

Abstract

Recent years witnessed deep learning models becoming more and more hungry for high-quality medical images. Nevertheless, this poses many challenges and restrictions in terms of medical imaging: acquired medical images cannot be easily obtained openly as high-quality data due to the reasons related to privacy, ethics and others; The data for positive sample and negative sample are often imbalanced; inter-domain noise due to the reason of lighting, etc., not making acquired data consistent with the real world. For the tongue images, these problems "low data—biased data distribution—difficult inter-domain properties" become worse.

To address these issues, this paper attempts to synthesize a large number of high-quality tongue images using DCGAN and Diffusion Models to expand the training set and enrich features. Due to local details and visual perception inconsistency of the image generation by DDIM and that of the real images, this paper innovatively improves its loss function. And as we compare the tongue images of healthy individuals and those of CRC patients, we specifically take multiple loss terms which include color histogram constraint, Structural Similarity (SSIM) loss and Perceptual Loss and perform parameter optimization on the improved model by a series of optimized experiments. The above methods try to preserve the shape consistency and texture details as well as the visual structure of tongue images more than before, so that the generated images could even closer reflect the visual characteristics of ground truth.

In order to verify the validity of this method, this article takes the tongue picture obtained from the hospital as the test set for the training. At the same time, the generation effect under the different loss function was compared, the results showed that the DDIM with improved loss function (Tongue-DDIM) is far superior to the traditional DCGAN model in terms of detail expression degree and texture authenticity, and better than DDIM without improved loss function. Later tongue image dataset generated, Colorectal Cancer (CRC) classification experiments demonstrated the enhancing of the subsequent classification accuracy and the complete performance of classification, for each CRC sample from the subsequent dataset. The research in this paper verifies that medical image generation based on diffusion generative models can be realized for tongue image data. It constructs a deep generative model optimized for the characteristics of the tongue image domain, which can further provide high-quality data support for the clinical diagnosis of CRC and explore a technological path for non-invasive early screening of CRC.

Keywords: Deep learning, Medical images, Diffusion Models, Tongue-DDIM, DCGAN

目　录

第 1 章 绪论

1.1 引言

结直肠癌（Colorectal Cancer，CRC）的全球发病率排名第三，死亡率排名第二，全球每年确诊的病例已超过 190 万，其中 20～49 岁的早发结直肠癌在逐年上升，达到平均 3%～4%/年，在英格兰、智利、新西兰等国家更为显著。结肠镜和病理活检虽仍是目前公认的结直肠癌的确诊方法，但因其具有侵入性、昂贵、医疗资源不均衡等特点，使得人群开展大范围早期筛查难以实现 [1]。

中医舌诊自古有“舌为脏腑之镜”之说；现代多组学研究进一步揭示，舌苔菌落微生态与消化道肿瘤密切相关[2]。CRC 患者舌苔 Fusobacterium nucleatum 等菌群丰度显著升高，支持“口-肠轴”致癌假说[3]；利用 16S rDNA 高通量测序技术对 CRC 患者和健康对照的舌苔进行分析，结果显示 CRC 患者舌苔明显增厚且微生物群落结构与健康组存在显著差异，舌苔微生态可作为 CRC 的潜在生物标志物之一。这种非侵入、可重复的可视窗口让“拍照即预筛”成为可能。利用 AI 技术，可以利用舌象，将这种非侵入性早筛大规模普及，为 CRC 的临床诊断提供帮助。但公开舌象图像数据集规模仅数百张（最广引用的公开集仅 668 幅标注舌像）[4]，且患病与健康样本严重失衡，远不足以训练具有临床泛化能力的深度模型。

为突破“少样本+分布失衡”瓶颈，本文引入生成式 AI（AIGC）技术，对抗网络（GAN）和去噪扩散模型（DDPM）。GAN 已在肝脏、皮肤等领域验证可显著提升少样本分类准确率[5]。最新 DDPM 架构在医学影像合成中表现出更高保真度与更广模式覆盖[6]。通过构建高保真舌象图像合成框架并与 CRC 早筛网络耦合评估，可在不增加采集负担与隐私风险的前提下扩充多样化训练样本，为低成本、可复制的非侵入式早期检测路径奠定数据与算法基础。

1.2 研究背景

1.2.1 结直肠癌流行趋势与筛查困境

全球范围来看，结直肠癌（CRC）已经成为死亡率和发病率最高的恶性肿瘤之一，据 WHO 统计，2020 年全球 CRC 新发病例及死亡人数均高达 190 万例与 93 万例，分别占全球癌症种类的第三位与第二位。2040 年全球 CRC 新增病例数与死亡人数预计将分别增至 320 万例与 160 万例，较目前的水平增长 63%与 73%[7]。中国罹患 CRC 的疾病负荷日益增加。根据 2019 年 GBD 统计数据，中国 CRC 的新发病例数约为 60.8 万例，约

占全球的 30%。1990—2019 年，中国 CRC 的发病率平均增长率为 3.11%、死亡率增长率为 1.05%[8]。

全球 CRC 的发病率及死亡数呈现明显的空间分布差异性，这一差异主要源于筛查的可及性，高收入国家筛查率高，其 CRC 发病率的下降就是必然的，然而绝大多数中低收入国家的 CRC 发病率是增加的。因此，在多国家开始探索生产起来廉价、使用起来安全而适合大规模推广的早诊早查的新技术，如粪便免疫化学试验（fecal immune chemical test，FIT）、粪便 DNA（mt-sDNA）等无痛、安全的粪便检测在一些国家和地区已获得积极回报[9]。而作为 CRC 患者最多的中国迫切需要制定适合中国现状的早诊早查方法，加大力度防控中国 CRC 的公共卫生问题。

1.2.2 中医舌诊及其现代科学依据

舌诊发源于《黄帝内经》，主张“舌者，外候之可征，脏腑之鉴”。现代组学的研究证实，舌苔微生态与肠道 CRC 紧密相关。Han 等通过 16SrRNA 高通量测序研究分析结直肠癌(CRC)患者以及健康对照的舌苔发现，结直肠癌患者的舌苔更增厚，而且其微生物群的分布规律与健康对照组之间的差异显著，舌苔的微生态特征可能是 CRC 的预测新生物标记之一[10]。Xu 等人在宏基因组枪法测序中纳入 30 例 CRC 患者、30 例结肠直肠腺瘤和 30 例健康对照，并采用随机森林模型检测出 Atopobiumrimae、Streptococcussanguinis、Prevotellaoris 为关键菌株，且 AUC 为 0.915 成功区分 CRC 和对照，进一步确证舌苔的微生物群的诊断预测价值[11]。近期，一篇多中心横断面研究对 377 名受试者的舌像及舌苔的微生物进行了定量分析，发现“厚苔”“腻苔”及“剥苔”等舌像组分在 CRC 组差异有统计学意义($p<0.05$)以及舌色 Lab 值的变化与进展相关，这为定量化、标准化舌诊提供了实验基础[12]。并且，已有胃癌多中心前瞻性的研究证明，基于 2D 的舌像深度学习模型在胃癌筛查方面，AUC 远高于传统血清学指标，验证了舌像影像特征在消化道肿瘤筛查方面的探索价值[13]。这些研究结果表明，基于舌象图像的 AI 分类模型是 CRC 筛查的“拍照即筛查”的“轻”选择。

1.2.3 舌象图像数据不足问题

相比于 CT、MRI 等主流的医学图像，开放的舌象图像数据非常有限，迄今为止最大的标注数据集只有 668 张。这主要是因为舌象图像进行结直肠癌标注的金标准是肠镜结果检查结果，标注十分困难且人工成本高。另外相关法律对面部、口腔图像的机构间共享有一定的要求，使得临床数据的共享存在一定困难。小组内部的初步试验显示舌象图像的数据量不足对于模型的泛化能力是一个巨大的障碍。面对这个问题，已有的工作已经从多维度探索医学图像数据的扩充，下文中将对数据扩充进行具体介绍。

1.3 国内外研究现状及发展趋势

1.3.1 传统数据增强方法

针对不同医学影像数据集深度学习研究中数据集匮乏这一问题，在深度生成模型盛行之前，学者们常依赖传统的数据增强方式扩充医学影像数据。其中最简单的方式为图像翻转、旋转、裁剪、缩放、增亮、减暗等，这些操作能够在一定程度上提高医学影像数据集的多样性，降低过拟合。传统增强方法实现简单有效，也是增强模型鲁棒性的有效方式。但是这类方式生成的图像都是以原始数据为模板的变化图像，无法生成全新的病灶表现，仅局限在样本的变化范围内。

1.3.2 基于 GAN 模型的医学图像合成研究

然而，通过几何变换来增强数据集对于样本的分布的丰富性帮助不大。为了解决这一问题，生成对抗网络（GAN）因其能够学习训练数据分布，从噪声中生成逼真的新图像，自 2014 年提出以来就在医学影像合成中得到广泛关注。例如，在皮肤病变合成方面，Bissoto 等使用 Pix2Pix GAN 在 ISIC 皮肤数据上取得了 84.7%的病变分类性能[14]。Qin 等采用 StyleGAN 生成高质量皮肤病变图像，使分类准确率达到约 95.2%[15]。这些结果表明，通过 GAN 合成的图像可有效提升皮肤癌分类模型的性能。

1.3.3 基于 Diffusion 模型的医学图像合成研究

近来利用扩散模型（Diffusion Models）进行医学图像的合成，其中 DDPM 是当前除 GAN 之外一种强有力的替代物，它通过在时间步长上去噪获得医学图像的多维分布。Fast-DDPM 通过将 DDPM 的时间步由 1000 降到 10，既促进了训练与采样的效率，又能保证训练和生成质量，因而大大降低了图像生成所花费的时间。此外，在皮肤病学方面，扩散模型已经被用于生成皮肤镜照片，Du 等人提出的扩散架构比以往的皮肤病分割技术对分割任务有性能提升[18]。Patcharapimpisut 等人在扩散模型的基础上，探讨了不同类型提示词的组合方式在生成皮肤病图像时可获得更好的多样性与真实性，进而使数据不平衡的平衡得以改善[19]。从以上分析中可以看出，扩散模型在各模态下的医学图像合成已经应用到多个任务中来应对数据稀缺场景。

1.4 本文主要研究内容

针对前文所述的带有结直肠癌标注的舌象图像数据集存在的严重的数据匮乏问题，本文旨在实现基于深度学习的舌象图像合成模型框架。如图 1-1 所示，本文将分别采用深度卷积生成对抗网络（DCGAN）及去噪扩散隐式模型（DDIM）作为基础模型，针对

任务需求预处理训练数据，并在训练与评估实验中逐渐改进模型损失函数，并最终与组内分类网络结合来验证合成样本对 CRC 分类模型的技术指标提升效果，从而为 CRC 的非侵入式早筛提供技术储备。本文将从原理论述、模型修改、实验验证三方面具体阐述舌象图像合成模型 Tongue-DDIM 的具体算法和实现功能。

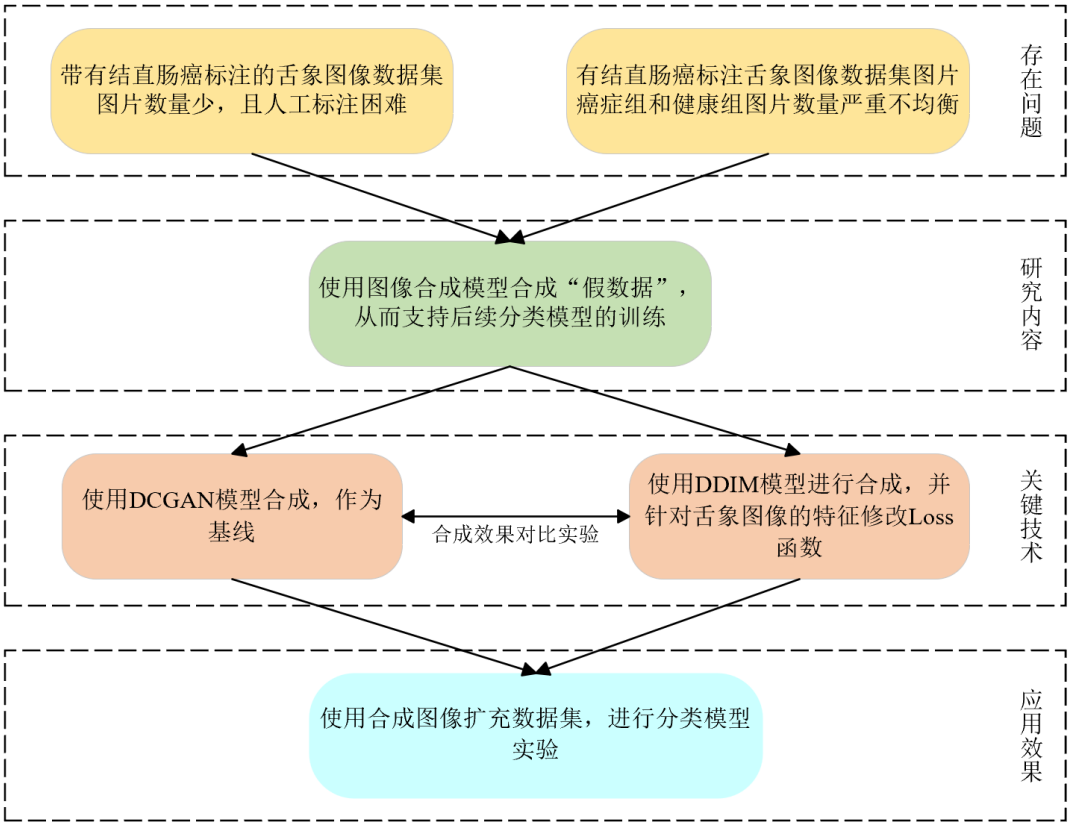


图 1-1 研究思路框图

第 2 章 研究任务需求分析及舌象图像数据集概况

本章主要介绍课题来源和舌象图像数据集存在的问题，解释本研究引入图像合成模型去扩充数据集的原因。另外，本章也介绍了本研究后续所用到的数据集的预处理工作。

2.1 基于舌象图像的结直肠癌分类诊断任务

本文模型生成的舌象图像所要应用的分类任务是基于现在的结直肠癌早期筛查实践需要，以中医舌诊理论和现代人工智能技术为核心手段，开发的舌象图像的结直肠癌分类诊断模型。具体地说，其应用 MedMamba[20]和 Vision Transformer[21]等分类模型，学习标注好的舌象图像特征，来自动对新的无标注舌象图像进行分类，即疑似结直肠癌和健康。

2.2 舌象图像数据集概况

本文所使用的数据集来自中山大学附属第六医院，使用舌面采集系统(DSO1-B)进行舌象特征采集。数据集标注由专业医生进行，标注标准如下：

（1）健康组：肠镜检查明确为正常者，或诊断为炎性增生性息肉/增生性息肉者；排除各类遗传性息肉病（家族性腺瘤性息肉病、P-J 综合征等）、肠道肿瘤相关性疾病、手术及化疗后患者及其他炎症性肠病、感染性疾病及非结直肠系肿瘤。

（2）患癌组：结、直肠癌或息肉恶变病理解证；除外非结、直肠上皮源性恶性肿瘤如神经内分泌肿瘤、淋巴瘤、肉瘤、肝癌等，接受过术后治疗（手术/化疗）影响舌像特征的患者。（说明：两组均需要排除重复情况，即同一个病人有多次录入的情况选择录入较早的一次进行分析；所有病人均需要有肠镜或病理证历诊断作为依据）

最终本文得到原始数据集：健康组 65 张，癌症组 228 张，均为 jpg 格式的图片。从数据数量即可看出原始数据集不足以支撑舌象图像分类诊断研究模型的训练，因此本课题研究的使用深度学习进行图像生成来增强数据集是后续研究的必要基础。

2.3 舌象图像合成任务数据集预处理

由于拍摄设备自动分割器的性能限制，有些图片含有牙齿等多余且干扰训练的信息. 如图 2-1 所示，本文使用 QuPath 软件将原始数据集手动分割为仅含有舌头主体部分，背景为黑色的标准图片。如图 2-3 所示，本文还撰写了 python 程序将所有图片自动转换为 128×128的尺寸，方便后续模型训练和生成。

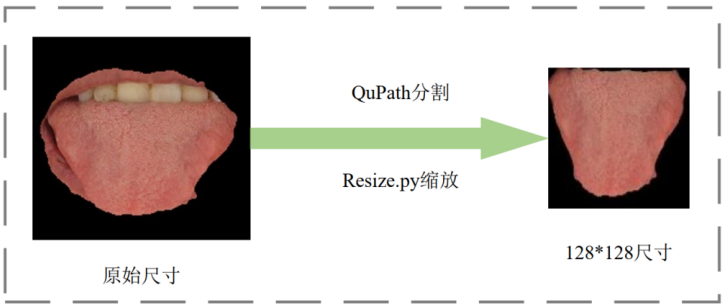


图 2-1 数据集预处理

由于图像合成模型训练所需数据数量亦不宜过小，如图 2-2 所示，本文又使用 python 程序，将处理好的数据集采用几何变换（随机旋转、随机平移、随机错切、随机缩放、随机水平翻转）的形式，将数据集扩充平衡。原始的健康组 65 张，癌症组 228 张先按照大约 8：1：1 的格式划分为训练集、测试集和验证集，然后再分别扩充。其中的训练集扩充至癌症组和健康组各 1100 张，用于合成模型的训练与分类模型实验。其中的验证集与训练集均分别扩充至癌症组和健康组各 130 张，这两个数据集仅用于后续的分类模型验证实验中。

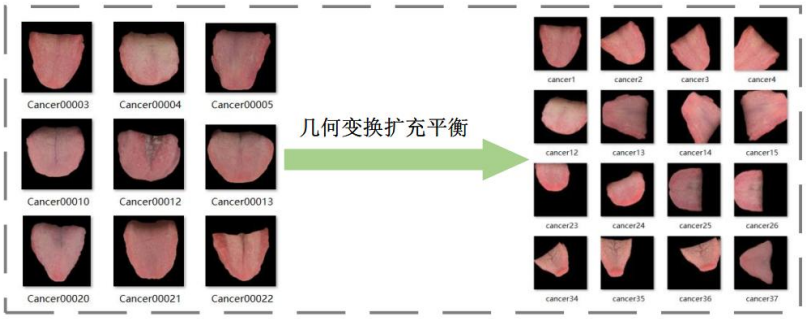


图 2-2 数据集几何扩充平衡

第 3 章 图像合成模型结构及算法

由于本研究主要着眼于使用 DCGAN 模型作为基线模型，同时将针对性地修改 DDIM 模型作为主要研究内容。本章将分别介绍 DCGAN 模型和 DDIM 模型的技术原理，以及 DDIM 某型相比 DCGAN 的技术优势。

3.1 DCGAN 的结构及算法

3.1.1 GANs 的核心结构

2014 年，Goodfellow 等首次提出生成对抗网络（Generative Adversarial Network，GAN）[22]，随后生成对抗网络以其强大的无监督学习能力，迅速渗透到包括图像生成、数据增强和一般性机器学习的其他领域。GAN 是后于一系列对抗生成模型（包括 DCGAN 等）的原理基础，其基本思路是基于博弈论中的两个人的零和博弈来构造两个对抗的神经网络模型—生成器（Generator）和判别器（Discriminator），从而在对抗中实现性能的提升。

如图 3-1 所示，GAN 的整体架构由两部分组成。生成器 G 的任务是学习真实数据的潜在分布，并从一个简单的先验噪声分布$p_z(z)$中采样，生成与真实数据尽可能相似的样本$G(z)$。判别器 D 则扮演着一个二分类器的角色，其目标是准确区分输入样本是来自于真实数据集$p_{\text{data}}(x)$还是由生成器 G 伪造的。判别器的输出$D(x)$表示样本 x 为真实数据的概率。

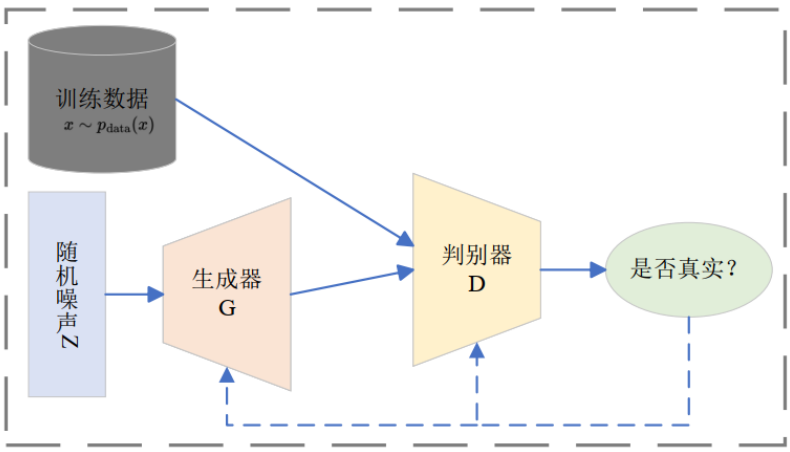


图 3-1 GAN 结构图

GAN 训练过程中存在一种动态的对抗过程，判别器 D 的目的是使其对真实样本和生成样本判断准确率尽可能地大，生成器 G 的目的是生成具有能够“欺骗”判别器能力

的样本，使判别器不能正确地对其产生的样本判断为真实样本，可用一个统一的价值函数 V(D，G)来描述二者的对抗过程，即：

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log\ D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log\,(1-D(G(z)))] \qquad (3\text{-}1)$$

公式中的 E 为期望。判别器 D 的目标是最大化 V(D,G)，而生成器 G 是最小化 V(D,G)。在具体训练中通常的优化办法是交替迭代训练，即先固定生成器 G 参数，梯度上升调整判别器 D 参数，使其更好地区分真实样本和生成样本；然后再固定判别器 D 的参数，梯度下降更新生成器 G 参数，使生成器 G 生成的样本越接近真实的数据分布，这样对判别器越“有效”。

3.1.2 DCGAN 在传统 GAN 上的改进

深度卷积生成对抗网络（DCGAN）将生成对抗网络（GAN）发展历程上的一个新高峰，DCGAN 通过引入卷积神经网络（CNN）强大的特征提取能力，有效增强了 GAN 生成图像质量和稳定性。Radford 等于 2015 年第一次系统性地描述了 DCGAN，并展示了 DCGAN 在无监督领域中突出的潜力[23]。本节对 DCGAN 网络结构及网络训练算法进行分析。

DCGAN 的设计满足如下的一些重要规则，其主要目的是解决传统 GAN 训练时不稳定的问题，如图 3-2 所示，重要的是使用深度卷积网络来构建生成器和判别器。对于生成器而言，DCGAN 没有像 GAN 所使用的全连接层、池化层，而是引入反卷积（deconvolution）来达到从低维噪声向量到高维像素空间的映射，进而让生成器学习自己的空间上采样技巧。具体来说，生成器的输入一般是随机噪声的向量 z，通过一系列的反卷积层和激活函数逐渐转化为目标图象的尺寸以及通道数。一个普通的生成器 G 可以定义为 G(z)，其目标是产生接近真实数据分布样本。

判别器 D 则更接近经典的卷积神经网络，它的责任是区分真实图像和生成器生成的伪图像。判别器的输入是一张图像，通过若干个卷积层和一个 Sigmoid 激活函数输出一个值在 0 与 1 之间，表示输入图像属于真实图像的可能性。判别器和生成器一样，没有采用池化层而是使用步幅卷积(stridedconvolution)进行下采样。另外，批归一化(BatchNormalization，BN)层广泛地应用在生成器和判别器的每层中，对于稳定网络训练过程、避免出现模式崩溃(modecollapse)、对梯度的传播都有好处。

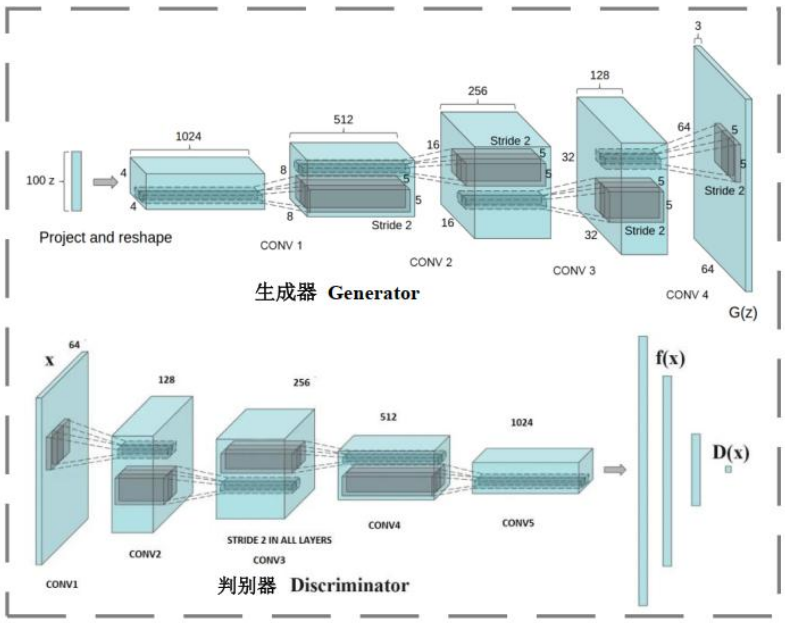


图 3-2 DCGAN 结构图

DCGAN 的成功不但因为其结构的改变，更是因为它证实了 GAN 框架能够在学习图像分布中成为一种可能，并为后续更好 GAN 模型的提出提供了良好的基础，它通过引入卷积结构和一系列稳定训练的策略而能够生成在视觉和语义上均良好的图片，在当时是一个重大的飞跃。

3.2 去噪扩散隐式模型（DDIM）的结构及算法

去噪扩散隐式模型（Denoising Diffusion Implicit Model，DDIM）是基于去噪扩散概率模型（DDPM）改进而来的，其大部分结构和核心思想都相同，只是在推理算法上做了优化。DDPM 是基于非平衡热力学的一类潜变量模型，并在高分辨率图像合成任务上取得了有突出的表现[24]。DDPM 使用参数化的马尔可夫过程通过变分推理来学习，使得通过固定的时间步长而产生近似真实数据分布的样本。DDPM 的核心思想是两个步骤，即一个固定不变的向前扩散过程以及一个学习的、反向的去噪过程。

3.2.1 DDPMs 的核心结构

如图 3-3 所示，DDPM 的结构可以理解为一个双向的马尔可夫链。

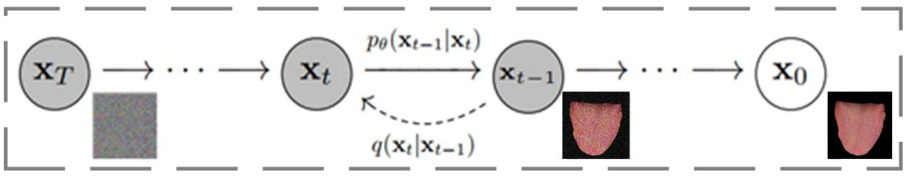


图 3-3 DDPM 的双向马尔可夫链结构图

在前向扩散(Forward Process/Diffusion Process)过程中，原始数据x_0通常为图像）通过 T 个时间步逐渐加入高斯噪声，直至数据几乎完全变成纯噪声。这个过程由一个预设的方差调度$\beta_1, \ldots, \beta_T$控制。每一步的转换$q(x_t \mid x_{t-1})$定义为：

$$q(x_t \mid x_{t-1}) := \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right) \tag{3-2}$$

其中$\mathcal{N}$表示高斯分布，I是单位协方差矩阵。一个重要的特性是，给定初始数据x_0，任意时刻 t 的含噪样本x_t可以通过以下闭式解直接采样得到：

$$q(x_t \mid x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I\right) \tag{3-3}$$

反向去噪过程(Reverse Process)则学习从纯噪声$x_T \sim \mathcal{N}(0, I)$开始，逐步去除噪声，最终生成目标数据$x_0$。这个过程同样是一个马尔可夫链，其转移概率$p_\theta(x_{t-1} \mid x_t)$也被参数化为高斯分布：

$$p_\theta(x_{t-1} \mid x_t) \coloneqq \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right) \tag{3-4}$$

当扩散过程中加入的噪声β_T足够小时，反向过程的转移同样可以设置为条件高斯分布，这使得神经网络参数化变得简单。

3.2.2 DDIM 在 DDPM 上的改进

DDPM 在图像生成领域取得了显著成果，但其采样过程通常需要模拟数千步的马尔可夫链，导致训练效率低，耗费时间极多。为解决此问题，DDIM 应运而生[25]。

如图 3-4 所示，DDIM 作为一种更高效的迭代隐式概率模型，DDIM 的核心创新在于将 DDPM 中马尔可夫性质的前向扩散过程推广到非马尔可夫过程。

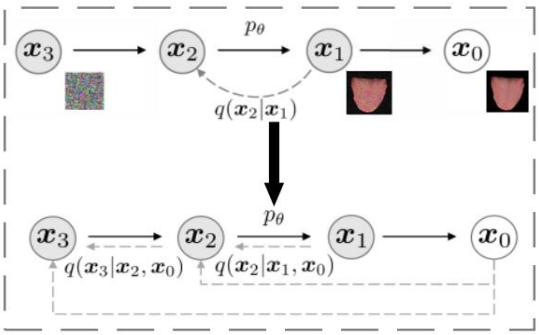


图 3-4 从 DDPM 的马尔可夫扩散过程推广到非马尔可夫过程

DDIM 考虑了一类更广泛的前向过程，这些分布虽然是非马尔可夫的，但其边际分布$q(x_t \mid x_0)$与 DDPM 中的保持一致。具体而言，DDIM 定义了一个新的非马尔可夫前向过程$q_\sigma(x_{1:T} \mid x_0)$，其反向条件分布$q_\sigma(x_{t-1} \mid x_t, x_0)$依赖于$x_t$和$x_0$：

$$q_\sigma(x_{t-1} \mid x_t, x_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}x_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\frac{x_t-\sqrt{\alpha_t}x_0}{\sqrt{1-\alpha_t}}, \sigma_t^2 I\right) \quad (3\text{-}5)$$

其中α_t对应 DDPM 中的$\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$，而$\sigma_t$是一个控制前向过程随机性的参数。当$\sigma_t \to 0$ 时，给定x_0和x_t，x_{t-1}变为确定性的 。

基于这种非马尔可夫前向过程，DDIM 设计了相应的生成过程。在生成（采样）阶段，给定x_t，模型首先预测去噪后的数据$f_\theta^{(t)}(x_t) \approx x_0$，然后利用该预测值通过上述反向条件分布$q_\sigma\left(x_{t-1} \mid x_t, f_\theta^{(t)}(x_t)\right)$来采样$x_{t-1}$。采样步骤可以表示为：

$$x_{t-1} = \sqrt{\alpha_{t-1}}\underbrace{\frac{x_t-\sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\alpha_t}}}_{\text{predicted } x_0} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2}}_{\text{coefficient of noise from } x_t}\cdot\epsilon_\theta^{(t)}(x_t) + \underbrace{\sigma_t\epsilon_t}_{\text{random noise}} \quad (3\text{-}6)$$

其中$\epsilon_\theta^{(t)}(x_t)$是预测的噪声。特别地，当所有$\sigma_t$=0 时，上式中无随机噪声项，生成过程变为确定过程。这使得 DDIM 转化为一种隐式概率模型，即可以由潜变量通过固定过程进行采样得到样本，与 GAN 类似。这样的确定性采样过程是 DDIM 相较于 DDPM 在算法思想上最大的不同，它可以在更少的采样步骤中生成高质量的样本，可以大幅度提高采样效率。

DDIM 最重要的一点便是其对前向扩散过程的泛化，使得采样更为自由且有效率。DDPM 是基于确定的马尔可夫过程的，而 DDIM 使用了一个非马尔可夫的前向过程，其核心是确保在任意时间点 t 含噪数据相对于初始数据的边际分布和 DDPM 中的一样，因而 DDIM 能够采用和 DDPM 完全一致的训练模型。在此框架下，DDIM 通过引入可调参数σ_t控制每步生成的随机性。当所有σ_t为 0 时，DDIM 的生成过程转为完全确定性，即从同一初始潜变量x_T出发将始终生成唯一输出样本x_0，这与 DDPM 因随机性导致从相同x_T生成不同样本的情况形成对比。这种确定性采样路径使采样速度提升了 10 至 50 倍。DDIM 的确定性生成过程赋予了其优良的样本一致性：即使用不同长度的采样轨迹，若起始潜变量x_T相同，生成样本的高级语义特征便能保持高度相似。这一特性使得 DDIM 能够直接在潜空间x_T中进行平滑且具语义意义的图像插值，此为 DDPM 因其固有随机采样过程而难以有效实现的。因此，DDIM 的创新主要通过泛化前向过程赋予采样阶段更大灵活性和效率，使预训练的 DDPM 模型焕发新生。

3.2.3 DDIM 中 U-Net 的结构与实现

DDIM 最关键的目标，就是学习一个函数（一般是神经网络）来估计每次去噪需要去掉多大程度的噪声，或者等价地去估计去噪后数据。如图 3-5 所示，DDIM/DDPM 的作者们使用了一个基于 U-Net 的神经网络实现这一关键的预测噪音的功能。

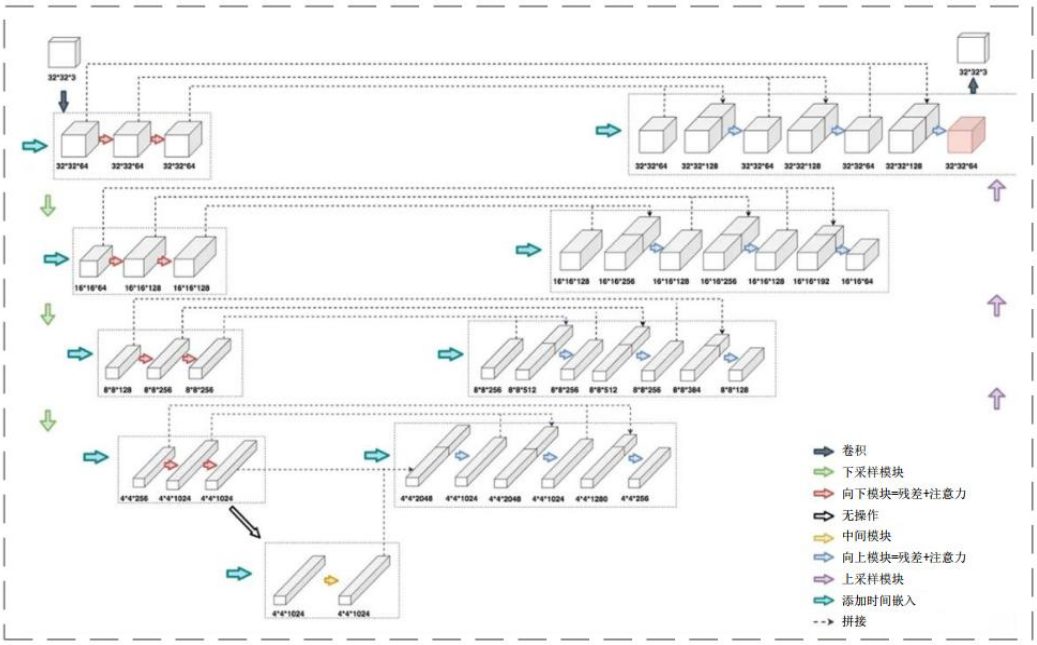


图 3-5 DDIM 的 U-Net 架构图

最初的 U-Net 是基于该编码器-解码器结构及其跳跃连接（skip connections）而在生物医学图像分割领域而崭露头角，且非常适合图像到图像的映射任务（也适用与 DDIM 中的根据噪声图像预测噪声或者生成去噪图像的任务），但是跳连接也是重要的特点。DDIM 中采用的 U-Net 骨架是类似于一种没有掩码的 Pixel CNN++。Pixel CNN++本身是一种具有强大自回归建模能力的模型且网络结构设计合理，能捕捉图像的局部依赖和全局依赖。从中汲取灵感设计了 U-Net 骨架的细节。下面对 U-Net 中的一些详细特征进行描述：

（1）归一化层（Normalization）：整个 U-Net 网络中广泛使用了组归一化。组归一化将通道分成若干组，并在每组内进行归一化。相比于批归一化（Batch Normalization），组归一化对批量大小不那么敏感，这在训练可能需要不同批量大小的生成模型时更为鲁棒。

（2）时间步嵌入（TimeEmbedding）：由于噪声预测函数既依赖于含噪图像也依赖于当前时间步 t，因此需要将时间信息有效嵌入网络。研究者们采用在 Transformer 中提出的正弦位置编码的方法来嵌入时间步 t。这样，对于不同的时间步 t，就可以获得不同

的、平滑地变化的表示向量，网络通过这种特殊嵌入能够感知不同的噪声水平。这些时间嵌入被添加至网络中间层的特征。

（3）参数共享：为使模型简单有效，U-Net 的时间步共享参数，在参数共享情况下，即整个网络在不同时间步使用相同的网络去噪，每个时间步的输入即为用于告知当前网络处于去噪的哪个阶段。

（4）自注意力机制（Self-Attention）：为获取图像中的长距离依赖信息，提高生成样本的全局一致性以及局部细节，研究者们在 U-Net 的较低的分辨率特征图上增加自注意力层。自注意力机制使网络在计算一个位置的响应时，直接用到了图像中的任意其他位置的特征，这对于生成复杂结构和纹理的图像来说是非常重要的。

DDIM 中的 U-Net 模型是 PixelCNN++的骨架、GroupNormalization、Transformer 式的时间嵌入及 Multihead 自注意力机制的组合，其核心是为了增强和优化 U-Net 结构来获得一个强大的且高效的噪声预测函数，其能够根据输入的观察到的含噪图像和当前的时间步，有效地估计出需要移除掉的噪音部分。

3.2.4 DDIM 与 GANs 的对比

在图像生成质量上，DDPM 优于当前的 GANs，比如在 CIFAR10 上获得很好的 InceptionScore(9.46)和 FIDScore(3.17)。这是因为相对于 GANs 来说，DDPM 是基于最大化似然的训练，训练平稳，无需应对 GAN 训练时易出现的模式崩溃问题和梯度问题，并且减少了 GAN 训练所需的繁琐超参数调参工作。因为 DDPM 是迭代去噪生成样本，生成的多样性比 GANs 中的模式崩溃生成多得多。DDPM 从去噪角度生成样本的过程更直观，也更容易分析。并且我们证明在适当参数条件下，DDPM 与去噪的分数匹配，退火朗之万的动能采样是等价的，也部分归因于 DDPM 的性能表现。另外，DDIM 修复了 DDPM 采样耗时、训练成本高的缺陷。由此，DDIM 具有训练高效、推理高效并且生成效果好的特点。所以，DDIM 模型也成为大家关注的研究点。

第 4 章 舌象图像去噪扩散隐式模型（Tongue-DDIM）设计

本章介绍本研究原版 DDIM 的基础上针对舌象图像合成任务做的针对性损失函数修改。本章还将介绍 Tongue-DDIM 在具体代码实现时的代码结构和参数配置。

4.1 Tongue-DDIM 模型损失函数设计

4.1.1 原始 DDIM 的损失函数

尽管 DDIM 引入了非马尔可夫前向过程和确定性采样路径，但其训练过程与 DDPM 完全相同，即优化相同的变分下界（Variational Lower Bound, VLB）。Ho 等进一步简化了这个目标函数，并提出了一种参数化方法，使得网络$\epsilon_\theta(x_t, t)$学习预测施加在x_0上的噪声ϵ。DDIM 的简化训练目标为：

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,x_0,\epsilon}\left[\left\|\epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right)\right\|^2\right] \quad (4\text{-}1)$$

其中 t 在{1,...,T}中均匀采样。

DDIM 的关键在于证明了其提出的非马尔可夫前向过程q_σ所对应的变分目标函数J_σ，对于特定的权重γ，等价于 DDPM 的损失函数L_γ（形式如公式(5)）加上一个常数 C。如其论文中的定理 1 所述：对于所有$\sigma > 0$，存在$\gamma \in \mathbb{R}_{>0}^T$和$C \in \mathbb{R}$使得$J_\sigma = L_\gamma + C$。这意味着，尽管 DDIM 的生成过程可以是非马尔可夫的，甚至可以是确定性的，但其训练目标与 DDPM 共享，因此可以直接使用预训练的 DDPM 模型进行 DDIM 采样，无需重新训练。

4.1.2 Tongue-DDIM 的损失函数

如图 4-1 所示，本文任务中的舌象图像是一类很特殊的数据。其背景是纯黑色，舌体本身呈现出自然的肉红色调，表面有一些纹理和细小的突起，即舌苔和其他舌象。这类图像的颜色区域相对集中，主要在红色、粉红色和少量黄白色的范围内变化，不像自然风景或复杂场景那样具有广泛而多样的颜色分布。

图 4-1 舌象图像数据案例图

通过本文进行的一些早期实验发现，舌象图像这样的颜色场景单一的图片会严重制约 DDIM 的合成效果。如图 4-2 所示，其合成结果表现为偏某一种颜色的彩色噪声问题。本文试分析产生这一结果的原因是颜色场景单一的图片在 RGB 三种颜色通道上的分别去噪，导致三种颜色通道去噪效果不均匀。而具有广泛而多样的颜色分布自然场景图片则在三个通道上都有均匀的信息量，所以不存在这个问题。

图 4-2 有单色彩色噪声问题的合成结果图

基于这些对 DDIM 模型的理解和早期实验结果的分析，本文针对舌象图像的颜色和纹理特性，新增了 4 种辅助损失函数旨在克服标准噪声估计损失(L_{simple})在捕捉特定医学图像特征方面的局限性。L_{simple}主要关注像素级别的均方误差，虽然能够驱动模型学习数据分布，但在生成如舌象这类颜色信息和细微纹理至关重要的图像时，可能无法充分保证其临床应用所需的视觉保真度。

本文将引入新的针对性损失函数的 DDIM 模型成为 Tongue-DDIM。下面分别介绍 4 种新增损失函数的细节。

（1）颜色均值损失函数（Color Loss）：该损失函数记为L_{color}，它的目标是最小化预测图像$\hat{x}_0$和真实图像x_0在各个颜色通道上的空间均值之差。设B为批大小，C为颜色通道数，$mean_{H,W}(\cdot)$ 表示在图像高H和宽W维度上取均值。

$$\begin{aligned} \mu_{\hat{x}_{0,c}} &= \text{mean}_{H,W}\left(\hat{x}_{0,c}\right) \\ \mu_{x_{0,c}} &= \text{mean}_{H,W}\left(x_{0,c}\right) \\ L_{\text{color}} &= \frac{1}{B \cdot C}\sum_{b=1}^{B}\sum_{c=1}^{C}\left(\mu_{\hat{x}_0,b,c} - \mu_{x_0,b,c}\right)^2 \end{aligned} \tag{4-3}$$

在代码中，这是通过(mean_pred - mean_true).pow(2).mean(dim=1)实现的，并在批次上求均值。总损失为$L_{\text{total}} = L_{\text{simple}} + W_{\text{color}} L_{\text{color}}$，其中$W_{\text{color}}$是颜色损失权重。

L_{color}通过直接约束预测图像与真实图像在各颜色通道上的全局平均颜色，力求生成的舌象在整体色调上与真实舌象保持一致。对于舌象诊断而言，舌色（如淡红、红、绛、紫等）是关键的辨证依据。相较而言，L_{color}能够更直接地指导模型学习并恢复出这些有诊断意义的宏观颜色特征，其计算上简单、有利于控制整体色偏，并适合于舌象这种颜色变化范围比较集中的图像等；但可能忽视颜色的空间分布和局部细节变化。

（2）颜色直方图损失（Color Histogram Loss）：该损失函数记为$L_{colorzf}$，它的目标是在匹配预测图像$\hat{x}_0$和真实图像x_0的归一化颜色直方图。设 $hist(I_c, K)$ 表示图像I的第c通道的归一化颜色直方图，其中有K个 bins。

$$L_{\text{colorzf}} = \frac{1}{B \cdot C} \sum_{b=1}^{B} \sum_{c=1}^{C} \| \text{hist}\left(\hat{x}_{0,b,c}, K\right) - \text{hist}\left(x_{0,b,c}, K\right) \|_2^2 \quad (4\text{-}4)$$

在代码中，这是通过对每个样本和每个通道计算直方图，归一化后计算其平方差和，再进行平均。总损失为$L_{\text{total}} = L_{\text{simple}} + W_{\text{colorzf}} L_{\text{colorzf}}$，其中$W_{\text{colorzf}}$是颜色直方图损失权重。

L_{colorzf}在颜色均值损失的基础上，提供了对颜色分布更为细致的约束。通过匹配预测图像与真实图像的颜色直方图，该损失函数不仅关注颜色的平均值，更关注各种颜色出现的频率。这对于舌象中可能存在的苔色、瘀点、瘀斑等局部颜色特征的生成具有优势，因为这些特征的颜色占比对诊断同样重要。相较于L_{color}，$L_{colorzf}$能更好地捕捉颜色的统计分布特性，减少生成图像颜色分布的单一化或失真。但它依旧不包含空间信息，可能无法完全保证颜色在舌面特定区域的准确复现。

（3）感知损失（Perceptual Loss）：该损失函数记为$L_{perceptual}$，它的目标是利用预训练的深度卷积网络（VGG16，设其特征提取函数为 $\Phi(\cdot)$）来比较预测图像$\hat{x}_0$和真实图像x_0在特征空间的相似性。通常需要对输入进行特定的归一化。

$$L_{\text{perceptual}} = \frac{1}{B} \sum_{b=1}^{B} \left\| \Phi\left(\text{norm}\left(\hat{x}_{0,b}\right)\right) - \Phi\left(\text{norm}\left(x_{0,b}\right)\right) \right\|_2^2 \quad (4\text{-}5)$$

在代码中，Φ对应 vgg_feature_extractor，norm 是指 VGG 的标准化预处理。损失是在特征图的均方误差上计算的。总损失为$L_{\text{total}} = L_{\text{simple}} + W_{\text{perceptual}} L_{\text{perceptual}}$，其中$W_{\text{perceptual}}$是感知损失权重。

$L_{\text{perceptual}}$从更高维度的特征空间对图像相似性进行度量。它利用预训练深度学习模型（如 VGG）提取的特征，这些特征被认为能更好地捕捉人类视觉系统所关注的语义信息和纹理结构。对于舌象而言，舌体的形态、舌苔的厚薄与质感、以及可能的裂纹或齿痕等，均属于重要的感知特征。标准L_{simple}难以直接优化这些高阶特征。$L_{perceptual}$的引入，能够显著提升生成舌象的真实感和细节丰富度，使其在视觉上更接近真实舌象。与颜色损失相比，感知损失更侧重于结构和纹理，与 SSIM 损失在目标上有所重叠，但其基于深度特征的度量方式可能捕捉到 SSIM 基于局部统计量所无法涵盖的更抽象的图像内容。

（4）结构相似性损失（SSIM Loss）：该损失函数记为L_{ssim}，它的目标是基于结构相似性指数 (SSIM) 或多尺度结构相似性指数(MS-SSIM)来度量$\hat{x}_0$和x_0之间的相似度。SSIM 值越接近 1 表示越相似。

$$L_{ssim} = \frac{1}{B}\sum_{b=1}^{B}\left(1 - \mathrm{SSIM}\left(\hat{x}_{0,b}, x_{0,b}\right)\right) \quad (4\text{-}6)$$

在代码中，如果使用 MS-SSIM，则替换 SSIM 为 MS-SSIM。总损失为$L_{\text{total}} = L_{\text{simple}} + W_{\text{ssim}} L_{\text{ssim}}$，其中$W_{\text{ssim}}$是 SSIM 损失权重。

L_{ssim}在从亮度、对比度和结构三个方面综合评估图像相似性。这对于保持舌象的整体轮廓、表面起伏以及与背景的清晰边界至关重要。相对的，L_{ssim}更符合人类对图像结构失真的感知。在舌象生成中，这意味着生成的舌体形态更规整，舌面细节的结构关系更合理。与感知损失相比，L_{ssim}更侧重于图像的局部结构保真度，计算也相对更直接。而感知损失则更关注全局的、更抽象的“看起来像”的程度。

针对颜色信息丰富且纹理结构具有诊断意义的舌象合成任务，在 DDIM 原有的噪声估计损失基础上引入颜色均值损失、颜色直方图损失、感知损失和结构相似性损失，构成了一个多目标优化的框架。相比单一的 L_{simple}，它们能从多个维度引导模型生成更符合特定应用需求的舌象，有望在颜色准确性、纹理细节和整体真实感方面取得显著提升。

4.2 Tongue-DDIM 模型代码结构

在本研究当中，Tongue-DDIM 模型的代码实现是遵循着模块化的设计原则来开展的。其核心结构在借鉴了现有的、基于 PyTorch 的 DDIM 开源实现，但在其基础上做了进一步的扩展。

代码主要分为如下几个模块：配置管理：configs/，使用 YAML 格式对实验参数进行统一管理。数据处理：datasets/，舌象数据集的读取及适配，这里包含了针对舌象图像这种特殊模态的数据增强或者数据预处理的专用逻辑；核心功能函数：functions/，其中 losses.py 是实现模型 loss 部分，集成了本文所提出的专属于舌象合成任务的四类辅助 loss。模型在训练时可以通过配置文件来动态指定并组合 loss 项的参数，通过 loss_registry 完成；在该文件夹里 denoising.py 则是具体实现 DDIM 逆向的去噪采样过程；模型定义：models/，其中 Model 定义了 U-Net 作为扩散模型的基本架构；运行与管理：runners/和 runners/目录下的 main.py，diffusion_run.py 定义了 Diffusion 类，是整个实验流程的核心类，封装了整个扩散模型流程的生命周期，main.py 是整个流程的总入口，根据不同用户提供的命令(训练，采样，测试)调用不同的方法，从而开启整个流程。

4.3 Tongue-DDIM 模型参数设置

此节主要讲述模型在数据处理、网络架构、扩散流程以及训练策略等多个方面核心参数的设定情况。这些参数综合起来，对模型的学习表现以及最终的生成效能起重要影响。

在数据处理方面，模型会把输入图像的尺寸统一调整成 128×128 像素，接着将其当作标准的 3 通道 RGB 图像来处理。在配置文件里明确指定了数据集存放的路径所在。为了让模型的泛化能力得以增强，在训练期间采用了随机水平翻转这种数据增强的策略。输入图像的像素值经过重新的缩放处理，一般会被规范到[−1,1]这个区间当中，以此来契合网络激活函数的有效范围，同时保障训练的稳定性。

Tongue-DDIM 采用的 U-Net 主干网络的关键参数配置如下：输入与输出图像的通道数均为 3。U-Net 的初始卷积层输出特征图的通道数设定为 128。网络在不同深度层级的通道数通过 ch_mult: [1, 1, 2, 2, 4, 4]参数进行控制，这意味着随着网络深度的增加和特征图分辨率的降低，特征通道数会相应倍增。在每个分辨率层级，均堆叠了 2 个残差块以深化特征提取。为了捕捉更广阔的上下文信息，在特征图分辨率降低至 16×16 时，模型引入了自注意力机制。模型训练时启用了 EMA，其平滑衰减率设置为 0.999，旨在获得更稳定和高质量的生成模型。U-Net 中的上下采样操作均伴随卷积层，以学习更优的尺度变换。

扩散过程所涉及的参数对从清晰图像转变为噪声图像的前向加噪过程做出了界定：其中，beta_schedule 运用的是“linear”策略，这意味着噪声方差β_{t}会从其初始值β_{start}开始，直至终止值β_{end}呈现出线性增长的态势。在此过程中，β_{start}被设定成了 0.0001 的数值，与之相对的，β_{end}则被设定为 0.02。整个扩散流程的总时间步数设定为 1000 步。上述这些参数综合起来，确定了噪声引入的速度情况以及最终所形成噪声的强度程度，它们对于模型的学习环节以及生成出来的质量状况有着直接的影响作用。在优化器这块，模型选取的是 Adam 优化器来对参数进行更新操作。学习率方面则设置成了一个相对较小的数值 0.0002，以此来确保训练过程能够保持稳定的状态。

训练过程中的控制参数涉及多个方面。其中，训练时把批处理大小（batch_size）设定成了 8。计划开展的总训练轮数（n_epochs）规定为 500，与此同时，也能够凭借总迭代次数（n_iters）来加以控制。为能对训练进程予以监控并且留存中间所产生的结果，模型检查点的保存频率（snapshot_freq）被设置为每经过 5000 个训练迭代步就保存一次。在图像采样这个阶段，其默认的批处理大小（batch_size）是 1，即仅仅会把最后一步生成的图像保留下来。

上述参数的设定，是综合考虑了模型表达能力、训练收敛情况、计算资源耗费等方面，同时也是结合舌象数据自身特性做的初步考量后才确定下来的。具体而言，通过灵活地去调整配置文件里的 model.type 参数，再把它和 functions/losses.py 中所定义的诸多

损失函数以及它们各自相应的权重相结合起来，本文可以较为系统地对不同损失组合展开探索。

第 5 章 实验结果与分析

本章介绍舌象图像合成实验的相关结果与分析。本文实验使用的训练和推理设备是 Nvidia RTX2080Ti 显卡。代码运行在 Lunix 系统中 Anaconda 框架下的 PyTorch 环境。本文实验采用的训练集均是第二章所述的扩充平衡后的数据集，癌症组和健康组各 1100 张图片，128*128 尺寸，jpg 格式，三颜色通道。由于癌症组的舌象图像和健康组的舌象图像在特征细节上的不同，且后续分类实验中他们也所述不同标签，本文将对于这两种数据类型分别进行训练和推理。下面分别详细介绍合成实验的评价标准及两种舌象图像的合成实验。

5.1 舌象图像合成实验的评价标准

在客观评价方面，本文查询文献资料后，决定采用弗雷歇初始距离（Fréchet Inception Distance，FID）和初始分数（Inception Score，IS）来评价 DCGAN 模型和 Tongue-DDIM 模型在舌象图像合成任务上的效果。本文编写了 python 程序（eval.py）来实现分数计算。

FID 是一种广泛用于评估 GANs 及其他生成模型所生成图像与原始图像相似性的指标。它通过比较真实图像集和生成图像集在 Inception-v3 网络的某一激活层输出特征的统计特性来衡量两者之间的距离。具体来说，FID 计算这两个特征集的均值和协方差矩阵之间的 Fréchet 距离：

$$FID(x,g)=\left\|\mu_x-\mu_g\right\|_2^2+\mathrm{Tr}\left(\Sigma_x+\Sigma_g-2\left(\Sigma_x\Sigma_g\right)^{1/2}\right) \tag{5-1}$$

其中，μ_x和μ_g分别是真实图像和生成图像在 Inception 网络激活层输出特征的均值向量；Σ_x和Σ_g是对应的协方差矩阵；$Tr(\cdot)$表示矩阵的迹。FID 值越低，表示生成图像集的特征分布与真实图像集的特征分布越接近，即生成图像与原图相似性越高。FID 对噪声、模糊、模式缺失等问题较为敏感。

IS 借助一个在 ImageNet 大规模数据集上预训练的 Inception-v3 图像分类网络，旨在量化合成图像的两个核心特性：清晰度与多样性。其计算公式：

$$IS=\exp\left(\mathbb{E}_{x\sim p_g}[D_{KL}(p(y\mid x)\|p(y))]\right) \tag{5-2}$$

其中，x是从生成器p_g采样的图像；$p(y\mid x)$是 Inception 网络对图像x预测的类别条件概率分布；$p(y)=\mathbb{E}_{x\sim p_g}[p(y\mid x)]$是所有生成图像的边际类别概率分布；$D_{KL}$是 KL 散度。

IS 值越高，通常表示生成图像的质量越好且多样性越高。高清晰度意味着$p(y \mid x)$的熵较低，即网络对分类结果很确定。高多样性意味着边际分布 p(y) 的熵较高，即各类别的图像都能生成且分布均匀。

5.2 癌症组舌象图像合成实验结果与分析

本文首先使用了一个针对舌象图像任务修改后的 DCGAN 模型训练并合成了 1100 张舌象图像。具体来说，本文修改了 DCGAN 原代码中的输入层尺寸和训练样本格式，让其能够训练本文所需的数据集。如图 5-1 所示，DCGAN 合成的图像虽然有一定的原始图像的特征，但整体视觉上效果较差，舌象图形形状不完善，且有噪声问题。这一结果说明对于合成 128*128 大尺寸图像的任务，传统的 DCGAN 模型无法有效完成。其合成数据集对于后续的分类任务能起到的帮助也有限。

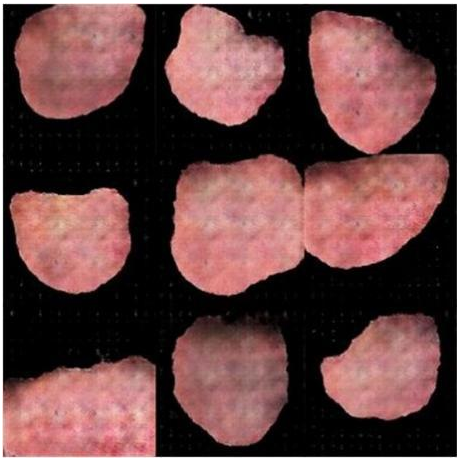


图 5-1 DCGAN 合成结果示意图

然后，本文采取第四章所详述的 Tongue-DDIM 模型进行实验。本文对第四章中所述的多种损失函数进行了训练，并保存了多种总迭代次数（n_iters）的模型权重进行效果对比。由于癌症组的图像特征丰富，信息含量多，推理时间步长取 150 步，以便于模型更好的去噪，获得与原数据集更相似的图片。本文选择其中的 35000 次迭代，40000 次迭代和 55000 次迭代进行了推理合成，每次合成 1100 张舌象图像。如图 5-2 所示，主观上看的 Tongue-DDIM 模型合成的图像在形状与色彩上比 DCGAN 的有很大进步。这一结果符合前文所说 DDIM 在图像生成质量上超越了 GANs。其合成数据集足以“以假乱真”，帮助后续的分类任务。

图 5-2 Tongue-DDIM 合成结果示意图

图 5-3 和图 5-4 展示了癌症组数据在各种损失函数版本的 Tongue-DDIM 在不同总迭代次数下的 IS 与 FID 得分。从中，本文分析出以下几个规律：

（1）Tongue-DDIM 合成结果比 DCGAN 显著性更好。从所有的三个记录点可以看出，所有 DDIM 变种的 FID 显著性低于 DCGAN，FID 值越低意味着生成的图像质量、真实度越高，生成图像的分布更加与真实数据分布接近；在 IS 上大多数 DDIM 变种表现均比或等于 DCGAN，IS 越大代表生成图像质量、清晰度与多样性越好。

（2）FID 会随着训练迭代次数上升而持续向好。随着训练迭代次数从 35000 到 55000 增加，除了 DINO3，其他所有 DDIM 变种 FID 值都是持续下降的，说明随着训练的进行模型生成结果在总体上稳步地得到提升。

（3）在训练次数变多时，IS 会波动或者略微下降后再变得更好。IS 在多个迭代次数上的表现不是一致的。IS 出现这种情况的原因可能是模型在学习中对于清晰度和多样性的权衡，在某个时刻，为了更好的拟合真实数据分布，可能会在某个 IS 衡量的方面做出一些牺牲或变化。

对于癌症组来说，由于其数据本身特征丰富，所含信息量较大。在进行数据扩充时，可以主要考虑将数据量提上来。因此，本文综合考虑 IS 所代表的多样性和 FID 所代表的相似性，本文选择优先考虑相似性即 FID，多样性即 IS 可以有所牺牲，但也要保证 IS 分数高于 DCGAN。最终选择 Simple Loss-55000 轮次合成的 1100 张癌症组舌象图像作为数据集扩充部分参与后续的分类任务。

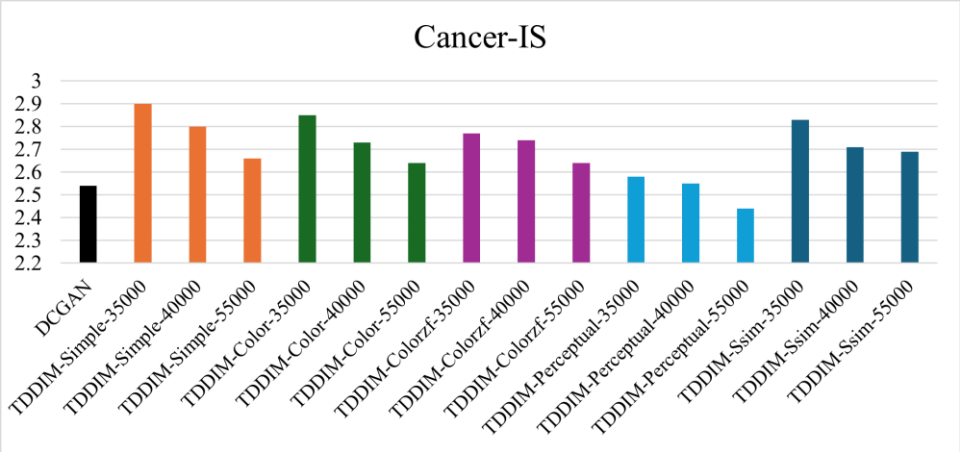


图 5-3 癌症组 Tongue-DDIM 合成结果 IS 分数柱状图

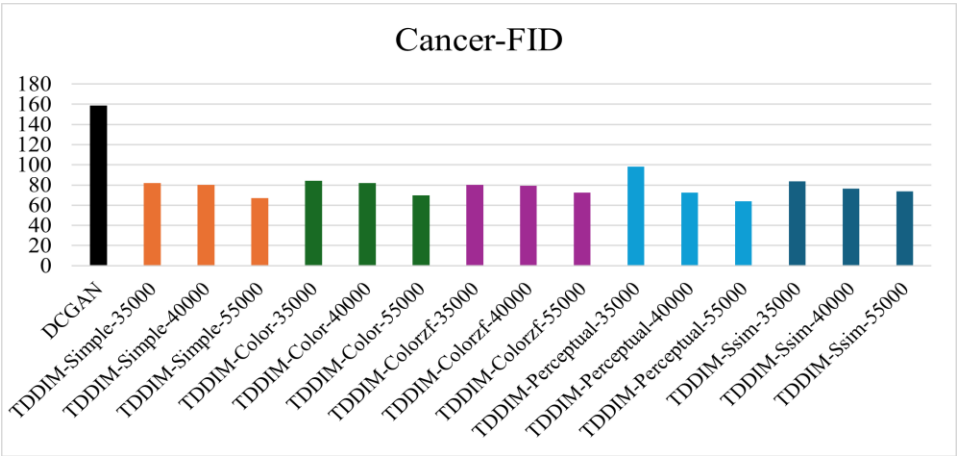


图 5-4 癌症组 Tongue-DDIM 合成结果 FID 分数柱状图

5.3 健康组舌象图像合成实验结果与分析

在 DCGAN 方面，健康组实验与上文的癌症组一致。在 Tongue-DDIM 方面，针对健康组的图像特点，记录 30000，40000，55000 三个迭代总数的权重。因为健康组的舌象舌苔特征等信息较少，如果推理时间步长过大或总迭代次数过大容易出现过拟合，造成 IS 显著低于 DCGAN 的问题。

根据图 5-5 和图 5-6 所示，健康组实验结果在与 DCGAN 比较方面和随迭代次数变化方面与癌症组保持一致。但是因为健康组本身含有的有效信息较少，本文在选择时优先考虑扩充数据集可以增强数据集整体的多样性，更有助于后续分类模型学习。健康组应更多考虑 IS 分数，因此选择 Ssim Loss-55000 轮次合成的 1100 张健康组舌象图像作为数据集扩充部分参与后续的分类任务。

图 5-5 健康组 Tongue-DDIM 合成结果 IS 分数柱状图

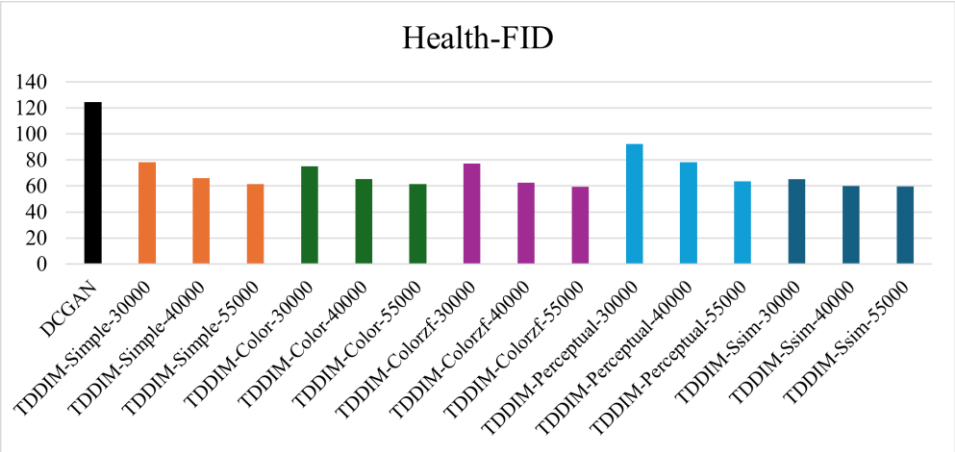


图 5-6 健康组 Tongue-DDIM 合成结果 FID 分数柱状图

5.4 舌象图像分类实验结果与分析

为进一步验证本研究提出的 Tongue-DDIM 模型所合成舌象图像的有效性及其在提升下游任务性能方面的潜力，本节将合成数据应用于舌象图像分类任务，并对实验结果进行分析。实验选取了 Medmamba 与 Vision Transformer (ViT) 两种主流的深度学习模型作为分类器，分别在仅使用真实数据集、使用真实数据辅以 DCGAN 合成数据扩充以及使用真实数据辅以本研究的 DDIM 合成数据扩充的三种条件下进行训练和测试。评估指标包括准确率（Accuracy）、F1 分数（F1-Score）、特异性（Specificity）、敏感性（Sensitivity）以及 ROC 曲线下面积（AUC）。

实验结果如表 5-11 所示，可以从中总结出以下几个规律：

（1）不管是应用 DCGAN 生成的数据还是 DDIM 生成数据对原始真实数据进行数据增强，这两类分类器的分类性能在各项评估指标上均存在较好的提升。从这个方面完全证实了应用合成数据缓解医学图像中数据集稀缺问题，提升分类模型泛化性能的有效性。

（2）Tongue-DDIM 生成的训练数据更有利于增强分类性能。对 DCGAN 与 Tongue-DDIM 生成的补充数据作对比分析，可见对两种分类器而言，一般 Tongue-DDIM 生成补充的数据较 DCGAN 生成补充的数据分类性能增益更明显。对 Medmamba 分类器而言，Tongue-DDIM 补充数据的 Accuracy(0.758)优于 DCGAN 补充数据 Accuracy(0.731)。对 ViT 分类器来说，Tongue-DDIM 补充数据的效果更为明显，Tongue-DDIM 补充数据的 Accuracy(0.777vs0.715)，Specificity(0.915vs0.669)，AUC(0.839vs0.829)较 DCGAN 补充数据的效果更为显著。

表 5-11 舌象图像分类实验结果

Model&Data	Accuracy	F1-Score	Specificity	Sensitivity	AUC
Medmamba 原数据集	0.669	0.739	0.400	0.938	0.702
Medmamba：DCGAN 扩充	0.731	0.762	0.600	0.862	0.871
Medmamba：TDDIM 扩充	0.758	0.753	0.777	0.738	0.827
Vit 原数据集	0.650	0.565	0.846	0.454	0.748
Vit：DCGAN 扩充	0.715	0.728	0.669	0.762	0.829
Vit：TDDIM 扩充	0.777	0.741	0.915	0.638	0.839

这一结果表明，本研究的 Tongue-DDIM 模型以及所提出的损失函数更能生成质量更高的、包含更多的信息、更接近真实数据分布的舌象图像，更能为后续分类任务提供有更有效的训练数据。Tongue-DDIM 生成的模拟图像的高保真度与多样性可能是其可以更好地模拟真实数据进而能够获得分类器学习更为鲁棒的判别特征的表现。

结　论

对于 CRC 的早期筛查，特别是低成本的、非侵入式筛查方式对于提高患者的生存率至关重要。本文从基于深度学习的舌象图像—CRC 分类诊断这一课题出发，试图解决有 CRC 标注的的舌象图像存在的数据量不足、不同类别数据不均衡问题。本文对基于深度学习的图像合成方法进行研究，利用深度卷积生成对抗网络（DCGAN）和去噪扩散隐式模型（DDIM）两种方法来合成高质量的、具有较强泛化性的伪造舌象图像，用以扩充 CRC 分类诊断模型所需的训练样本，丰富其数据集特征，成功提高了后续 CRC 分类诊断模型的精准性。

本文主要思想是对 DDIM 模型中的损失函数进行改进，结合舌象图像存在的颜色特征和结构特征，引入颜色直方图惩罚损失函数、结构相度损失函数及感知损失函数，构建了 Tongue-DDIM 模型，实验表明，Tongue-DDIM 相比传统 DCGAN 模型，在合成舌象图像的细节特征和纹理真实性等主观感受上，以及 FID 和 IS 等客观指标上都有显著优势；在将 Tongue-DDIM 输出的仿真舌象图像进行 CRC 分类，可以提升分类模型分类精度和综合性能。

综上所述，本文实现了去噪扩散图像合成模型在医学图像中的合成应用，验证了该技术在舌象数据集的实践应用可行性与有效性。文中所提出 Tongue-DDIM 模型针对舌象图像特点设计，可合成高清仿真样本，为基于深度学习技术的 CRC 临床分期诊断以及非侵入早期筛查提供数据集扩充的基础。本文为 CRC 研究提供了一种低成本非侵入基于 AI 的早期筛查临床诊断干预技术路径。

参考文献

[1] Chang S H, Patel N, Du M, et al. Trends in early-onset vs late-onset colorectal cancer incidence by race/ethnicity in the United States Cancer Statistics Database[J]. Clinical Gastroenterology and Hepatology, 2022, 20(6): e1365-e1377.

[2] 赵健翔, 曹龙飞, 付立群, 等. 口腔菌群与消化道肿瘤关系的研究进展[J]. Advances in Clinical Medicine, 2023, 13: 8347.

[3] Vogtmann E, Yano Y, Zouiouich S, et al. The human oral microbiome and risk of colorectal cancer within three prospective cohort studies in the United States[J]. Cancer, 2025, 131(6): e35802.

[4] Shi D, Tang C, Blackley S V, et al. An annotated dataset of tongue images supporting geriatric disease diagnosis[J]. Data in brief, 2020, 32: 106153.

[5] Frid-Adar M, Diamant I, Klang E, et al. GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification[J]. Neurocomputing, 2018, 321: 321-331.

[6] Ju Z, Zhou W. Vm-ddpm: Vision mamba diffusion for medical image synthesis[J]. arXiv preprint arXiv:2405.05667, 2024.

[7] Morgan E, Arnold M, Gini A, et al. Global burden of colorectal cancer in 2020 and 2040: incidence and mortality estimates from GLOBOCAN[J]. Gut, 2023, 72(2): 338-344.

[8] Li Q, Wu H, Cao M, et al. Colorectal cancer burden, trends and risk factors in China: a review and comparison with the United States[J]. Chinese Journal of Cancer Research, 2022, 34(5): 483.

[9] Schoenfeld P. Multi-Target Stool DNA Test for CRC Screening: How Accurate is the New Version?[J].

[10] Han S, Chen Y, Hu J, et al. Tongue images and tongue coating microbiome in patients with colorectal cancer[J]. Microbial pathogenesis, 2014, 77: 1-6.

[11] Chen Q, Huang X, Zhang H, et al. Characterization of tongue coating microbiome from patients with colorectal cancer[J]. Journal of Oral Microbiology, 2024, 16(1): 2344278.

[12] Liu F, Su D, Shi X, et al. Cross-population tongue image features and tongue coating microbiome changes in the evolution of colorectal cancer[J]. Frontiers in Microbiology, 2025, 16: 1442732.

[13] Yuan L, Yang L, Zhang S, et al. Development of a tongue image-based machine learning tool for the diagnosis of gastric cancer: a prospective multicentre clinical cohort study[J]. EClinicalMedicine, 2023, 57.

[14] Bissoto A, Perez F, Valle E, et al. Skin lesion synthesis with generative adversarial networks[C]//OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis: First International Workshop, OR 2.0 2018, 5th International Workshop, CARE 2018, 7th International Workshop, CLIP 2018, Third International Workshop, ISIC 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16 and 20, 2018, Proceedings 5. Springer International Publishing, 2018: 294-302.

[15] Shi Y, Abulizi A, Wang H, et al. Diffusion models for Medical Image Computing: a Survey[J]. Tsinghua Science and Technology, 2024, 30(1): 357-383.

[16] Sun Q, Huang C, Chen M, et al. Skin lesion classification using additional patient information[J]. BioMed research international, 2021, 2021(1): 6673852.

[17] Jiang H, Imran M, Ma L, et al. Fast denoising diffusion probabilistic models for medical image-to-image generation[J]. arXiv preprint arXiv:2405.14802, 2024.

[18] Du S, Wang X, Lu Y, et al. Boosting dermatoscopic lesion segmentation via diffusion models with visual and textual prompts[C]//2024 IEEE International Symposium on Biomedical Imaging (ISBI). IEEE, 2024: 1-5.

[19] Patcharapimpisut P, Khanarsa P. Generating Synthetic Images Using Stable Diffusion Model for Skin Lesion Classification[C]//2024 16th International Conference on Knowledge and Smart Technology (KST). IEEE, 2024: 184-189.

[20] Yue Y, Li Z. Medmamba: Vision mamba for medical image classification[J]. arXiv preprint arXiv:2403.03849, 2024.

[21] 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.

[22] Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets[J]. Advances in neural information processing systems, 2014, 27.

[23] Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks[J]. arXiv preprint arXiv:1511.06434, 2015.

[24] Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models[J]. Advances in neural information processing systems, 2020, 33: 6840-6851.

[25] Song J, Meng C, Ermon S. Denoising diffusion implicit models[J]. arXiv preprint arXiv:2010.02502, 2020.

附　录

附录 1：Tongue-DDIM 模型 Loss 程序（losses.py）

```
import torch


def noise_estimation_loss(model,
                          x0: torch.Tensor,
                          t: torch.LongTensor,
                          e: torch.Tensor,
                          b: torch.Tensor, keepdim=False):
    a = (1-b).cumprod(dim=0).index_select(0, t).view(-1, 1, 1, 1)
    x = x0 * a.sqrt() + e * (1.0 - a).sqrt()
    output = model(x, t.float())
    if keepdim:
        return (e - output).square().sum(dim=(1, 2, 3))
    else:
        return (e - output).square().sum(dim=(1, 2, 3)).mean(dim=0)


def noise_estimation_color(model,
                          x0: torch.Tensor,
                          t: torch.LongTensor,
                          e: torch.Tensor,
                          b: torch.Tensor,
                          color_loss_weight=1.0,  # 【修改】新增颜色损失权重参
数
                          keepdim=False):
    a = (1 - b).cumprod(dim=0).index_select(0, t).view(-1, 1, 1, 1)
    x = x0 * a.sqrt() + e * (1.0 - a).sqrt()
    output = model(x, t.float())

    # 原始噪声估计损失
```

```
    noise_loss = (e - output).square().sum(dim=(1, 2, 3))

    # 【新增】颜色一致性损失计算
    # 根据模型输出重构预测的 x0
    x0_pred = (x - (1.0 - a).sqrt() * output) / a.sqrt()
    # 计算预测图像和真实图像在空间维度上的 RGB 通道均值
    mean_pred = x0_pred.mean(dim=(2, 3))
    mean_true = x0.mean(dim=(2, 3))
    # 计算均值之间的 L2 损失（对每个样本，再对 batch 求均值）
    color_loss = (mean_pred - mean_true).pow(2).mean(dim=1)
    # 【结束新增】

    if keepdim:
        return noise_loss + color_loss_weight * color_loss
    else:
        return noise_loss.mean() + color_loss_weight * color_loss.mean()


def noise_estimation_colorzf(model: torch.nn.Module,
                             x0: torch.Tensor,
                             t: torch.LongTensor,
                             e: torch.Tensor,
                             b: torch.Tensor,
                             color_hist_weight: float = 1.0,
                             bins: int = 64,
                             keepdim: bool = False):
    # 根据扩散过程计算 a 值
    a = (1 - b).cumprod(dim=0).index_select(0, t).view(-1, 1, 1, 1)
    # 重构 x: 即加入噪声后的图像
    x = x0 * a.sqrt() + e * (1.0 - a).sqrt()
    # 模型输出噪声预测
    output = model(x, t.float())

    # 原始噪声估计损失
    noise_loss = (e - output).square().sum(dim=(1, 2, 3))
```

```
    # 根据模型输出重构预测的 x0
    x0_pred = (x - (1.0 - a).sqrt() * output) / a.sqrt()

    # 【新增】颜色直方图一致性损失
    # 对每个样本的每个通道计算直方图并计算二范数差异
    batch_size, channels, _, _ = x0.shape
    hist_loss_total = 0.0
    for i in range(batch_size):
        for c in range(channels):
            # 对预测图像和真实图像进行直方图计算（先将每个通道展平至 1 维）
            hist_pred = torch.histc(x0_pred[i, c].view(-1), bins=bins, min=0,
max=1)
            hist_true = torch.histc(x0[i, c].view(-1), bins=bins, min=0, max=1)
            # 归一化直方图
            hist_pred = hist_pred / (hist_pred.sum() + 1e-8)
            hist_true = hist_true / (hist_true.sum() + 1e-8)
            # 累计各通道直方图的差异损失（平方差和）
            hist_loss_total += (hist_pred - hist_true).pow(2).sum()
    # 平均到每个样本每个通道
    hist_loss = hist_loss_total / (batch_size * channels)

    if keepdim:
        return noise_loss + color_hist_weight * hist_loss
    else:
        return noise_loss.mean() + color_hist_weight * hist_loss



import torch
import torch.nn.functional as F
from torchvision import models

# 全局懒加载 VGG16 特征提取器（取前几层）
try:
    vgg_feature_extractor
```

```
except NameError:
    device = torch.device("cuda:5" if torch.cuda.is_available() else "cpu")
    vgg = models.vgg16(pretrained=True).features.to(device)
    # 例如取前 16 层（大致对应 conv3_3 层），可以根据需要调整
    vgg_feature_extractor                                                    =
torch.nn.Sequential(*list(vgg.children())[:16]).eval()
    for param in vgg_feature_extractor.parameters():
        param.requires_grad = False


def perceptual_loss(model: torch.nn.Module,
                    x0: torch.Tensor,
                    t: torch.LongTensor,
                    e: torch.Tensor,
                    b: torch.Tensor,
                    perceptual_weight: float = 1.0,
                    keepdim: bool = False):
    """
    Perceptual Loss 基于预训练 VGG16 特征提取器。

    参数说明：
      model: 用于噪声预测的模型（例如扩散模型）。
      x0: 原始图像，形状为 (B, C, H, W)，假设像素值在 [0, 1] 范围内。
      t: 当前时间步（扩散模型中的 t），类型为 LongTensor。
      e: 加入的噪声。
      b: 扩散过程中预定义的 beta 值序列。
      perceptual_weight: 感知损失部分的权重。
      keepdim: 如果为 True，则返回每个样本的 loss，否则返回平均 loss。

    实现思路：
      1. 根据 b 和 t 计算 a；
      2. 构造加入噪声后的图像 x；
      3. 使用模型预测噪声 output，并由此反解得到预测图像 x0_pred；
      4. 对 x0 与 x0_pred 进行 VGG16 归一化（VGG16 要求的均值与标准差）；
      5. 利用预训练的 VGG16 特征提取器得到特征，并计算它们之间的 L2 均方误差作
```

```
为感知损失；
    6. 最终 loss 由原始噪声估计损失与感知损失加权求和得到。
    """
    # 1. 计算 a，并重构 x
    a = (1 - b).cumprod(dim=0).index_select(0, t).view(-1, 1, 1, 1)
    x = x0 * a.sqrt() + e * (1.0 - a).sqrt()

    # 2. 噪声预测
    output = model(x, t.float())
    noise_loss = (e - output).square().sum(dim=(1, 2, 3))

    # 3. 根据预测输出反解 x0_pred
    x0_pred = (x - (1.0 - a).sqrt() * output) / a.sqrt()

    # 4. 对 x0 和 x0_pred 进行 VGG16 归一化：VGG 期望输入为 RGB 图像，
    # 均值和标准差分别为 [0.485, 0.456, 0.406] 和 [0.229, 0.224, 0.225]
    device = x0.device
    mean = torch.tensor([0.485, 0.456, 0.406], device=device).view(1, 3, 1, 1)
    std = torch.tensor([0.229, 0.224, 0.225], device=device).view(1, 3, 1, 1)
    x0_norm = (x0 - mean) / std
    x0_pred_norm = (x0_pred - mean) / std

    # 5. 利用 VGG16 特征提取器计算特征
    feat_x0 = vgg_feature_extractor(x0_norm)
    feat_x0_pred = vgg_feature_extractor(x0_pred_norm)

    # 6. 计算感知损失：对各样本特征的 L2 平方差再对空间维度取均值
    perceptual = F.mse_loss(feat_x0_pred, feat_x0, reduction='none')
    perceptual = perceptual.view(perceptual.size(0), -1).mean(dim=1)

    # 综合原始噪声估计损失与感知损失
    if keepdim:
        return noise_loss + perceptual_weight * perceptual
    else:
        return noise_loss.mean() + perceptual_weight * perceptual.mean()
```

```
from pytorch_msssim import ssim, ms_ssim

def noise_estimation_ssim(model: torch.nn.Module,
                          x0: torch.Tensor,
                          t: torch.LongTensor,
                          e: torch.Tensor,
                          b: torch.Tensor,
                          ssim_weight: float = 1.0,
                          use_ms_ssim: bool = False,
                          keepdim: bool = False):
    """
    结构相似性 (SSIM/MS-SSIM) 约束 Loss

    参数：
      model: 用于噪声预测的模型。
      x0: 原始图像，形状为 (B, C, H, W)，像素值范围假定在 [0, 1]。
      t: 当前时间步（扩散模型中的 t），类型为 LongTensor。
      e: 加入的噪声。
      b: 扩散过程中的 beta 序列。
      ssim_weight: SSIM 部分的损失权重。
      use_ms_ssim: 是否使用 Multi-Scale SSIM（若为 False，则使用标准 SSIM）。
      keepdim: 是否保持每个样本的 loss（True 返回每个样本的值，否则返回平均值）。

    实现思路：
      1. 根据 b 和 t 计算 a，并构造 noisy image x；
      2. 利用模型预测噪声 output，并根据输出反解 x0_pred；
      3. 计算 x0_pred 与 x0 的结构相似性值，定义 SSIM Loss 为 (1 - ssim_value)；
      4. 综合原始噪声估计损失与 SSIM Loss。
    """
    # 根据扩散过程计算 a
    a = (1 - b).cumprod(dim=0).index_select(0, t).view(-1, 1, 1, 1)
    # 构造加入噪声后的图像
    x = x0 * a.sqrt() + e * (1.0 - a).sqrt()
    # 模型预测
```

```
    output = model(x, t.float())
    # 计算基础噪声估计损失
    noise_loss = (e - output).square().sum(dim=(1, 2, 3))
    # 反解预测的 x0
    x0_pred = (x - (1.0 - a).sqrt() * output) / a.sqrt()

    # 计算 SSIM 或 MS-SSIM，相似性越高（数值越接近 1）越好，因此取 (1 - 相似性) 作为损失
    if use_ms_ssim:
        ssim_val = ms_ssim(x0_pred, x0, data_range=1.0, size_average=False)
    else:
        ssim_val = ssim(x0_pred, x0, data_range=1.0, size_average=False)
    ssim_loss = 1.0 - ssim_val

    if keepdim:
        return noise_loss + ssim_weight * ssim_loss
    else:
        return noise_loss.mean() + ssim_weight * ssim_loss.mean()

# 将新增的损失函数加入到损失注册表中
loss_registry = {
    'simple': noise_estimation_loss,
    'color': noise_estimation_color,
    'colorzf': noise_estimation_colorzf,
    'perceptual': perceptual_loss,
    'ssim': noise_estimation_ssim,
}
```

附录 2：Tongue-DDIM 模型评价程序（eval.py）

```
import os
import random
import numpy as np
from PIL import Image
import torch
```

```
import torch.nn.functional as F
import torchvision.transforms as transforms
import torchvision.models as models
from scipy.linalg import sqrtm

seed = 42
random.seed(seed)
np.random.seed(seed)
torch.manual_seed(seed)
torch.cuda.manual_seed_all(seed)

torch.backends.cudnn.deterministic = True
torch.backends.cudnn.benchmark = False
torch.use_deterministic_algorithms(True)



def load_images_from_folder(folder, transform=None):
    images = []
    for filename in os.listdir(folder):
        if filename.lower().endswith((".png", ".jpg", ".jpeg", ".bmp")):
            path = os.path.join(folder, filename)
            try:
                img = Image.open(path).convert('RGB')
                if transform is not None:
                    img = transform(img)
                images.append(img)
            except Exception as e:
                print(f"加载 {path} 时出错: {e}")
    return images

def calculate_inception_score(imgs, model, batch_size=32, splits=10,
device='cuda:2'):
    model.eval()
    preds = []
    imgs = torch.stack(imgs)
```

```
    with torch.no_grad():
        for i in range(0, len(imgs), batch_size):
            batch = imgs[i:i + batch_size].to(device)
            output = model(batch)
            if isinstance(output, tuple):
                output = output[0]
            preds.append(F.softmax(output, dim=1).cpu().numpy())
    preds = np.concatenate(preds, axis=0)

    split_scores = []
    N = preds.shape[0]
    for k in range(splits):
        part = preds[k*(N//splits):(k+1)*(N//splits), :]
        py = np.mean(part, axis=0)
        scores = [np.sum(pyx*(np.log(pyx+1e-10)-np.log(py+1e-10))) for pyx in
part]
        split_scores.append(np.exp(np.mean(scores)))
    return float(np.mean(split_scores)), float(np.std(split_scores))

def get_activations(imgs, model, batch_size=32, device='cuda:2'):
    model.eval()
    activations = []
    def hook(module, input, output):
        activations.append(output.cpu().detach().numpy())
    handle = model.avgpool.register_forward_hook(hook)

    imgs = torch.stack(imgs)
    with torch.no_grad():
        for i in range(0, len(imgs), batch_size):
            batch = imgs[i:i + batch_size].to(device)
            _ = model(batch)
    handle.remove()

    act = np.concatenate(activations, axis=0)
    return act.reshape(act.shape[0], -1)
```

```
def calculate_fid(act1, act2):
    mu1, sigma1 = np.mean(act1, axis=0), np.cov(act1, rowvar=False)
    mu2, sigma2 = np.mean(act2, axis=0), np.cov(act2, rowvar=False)
    diff = mu1 - mu2
    covmean, _ = sqrtm(sigma1.dot(sigma2), disp=False)
    if np.iscomplexobj(covmean):
        covmean = covmean.real
    return float(diff.dot(diff) + np.trace(sigma1 + sigma2 - 2*covmean))

def main():
    os.environ["KMP_DUPLICATE_LIB_OK"] = "TRUE"

    if not torch.cuda.is_available():
        raise RuntimeError("CUDA 不可用，请在支持 GPU 的环境中运行。")
    device = torch.device("cuda:2")
    print("使用设备：", device)

    inception_transform = transforms.Compose([
        transforms.Resize((299, 299)),
        transforms.ToTensor(),
        transforms.Normalize(mean=[0.485, 0.456, 0.406],
                             std=[0.229, 0.224, 0.225])
    ])

    gen_folder = ""
    real_folder = ""

    print("加载生成图片（Inception 预处理）...")
    gen_imgs = load_images_from_folder(gen_folder,
transform=inception_transform)
    print(f"共加载 {len(gen_imgs)} 张生成图片。")

    print("加载真实图片（Inception 预处理）...")
    real_imgs = load_images_from_folder(real_folder,
```

```
transform=inception_transform)
    print(f"共加载 {len(real_imgs)} 张真实图片。")

    if not gen_imgs or not real_imgs:
        print("图片数量为 0，请检查路径与格式。")
        return

    inception_model = models.inception_v3(pretrained=True,
transform_input=False, aux_logits=True)
    inception_model.to(device).eval()

    print("计算 Inception Score...")
    is_mean, is_std = calculate_inception_score(gen_imgs, inception_model,
batch_size=32, splits=10, device=device)
    print(f"Inception Score: {is_mean:.4f} ± {is_std:.4f}")

    print("提取真实图片特征...")
    act_real = get_activations(real_imgs, inception_model, batch_size=32,
device=device)
    print("提取生成图片特征...")
    act_gen = get_activations(gen_imgs, inception_model, batch_size=32,
device=device)

    fid_value = calculate_fid(act_real, act_gen)
    print(f"Frechet Inception Distance (FID): {fid_value:.4f}")

if __name__ == '__main__':
    main()
```

致　谢

时光荏苒，岁月如梭，四年的大学学习生涯即将画上句号。本毕业设计（论文）的顺利完成，离不开许多人的关心与帮助。在此，我谨向所有在我求学和论文撰写过程中给予我无私帮助的老师、同学、家人和朋友们，致以最诚挚的谢意。

首先，我要衷心感谢我的指导老师李兰兰教授。从论文的选题、开题、资料搜集、研究思路的确定，到实验过程的指导、论文框架的搭建，乃至最终成稿的修改与润色李李兰兰教授都倾注了大量的心血。她严谨的治学态度、敏锐的洞察力以及诲人不倦的师者风范，将成为我未来学习和工作中宝贵的财富。

同时，也要感谢同课题组的王子玥学姐。感谢她耐心地帮我解答一切在毕业设计期间遇到的问题，使得研究得以顺利进行。与她的交流讨论碰撞出了许多思想的火花令我受益匪浅。感谢她在论文格式与润色方面提供的无私帮助。她的学术素养与创新能力是我学习的榜样。

在论文写作期间，还要感谢我的舍友、朋友、家人在生活上对我的照顾与帮助。你们的理解和鼓励，是我能够安心完成学业、勇往直前的坚强后盾。感谢福州大学为我提供了良好的学习环境和丰富的学习资源。感谢物理与信息工程学院各位老师和行政人员的辛勤付出。由于本人学识水平有限，研究经验不足，论文中难免存在疏漏和不足之处，恳请各位老师和专家批评指正。再次向所有关心、支持和帮助过我的人们表示最诚挚的感谢！