|  |  |
| --- | --- |
| 实践作业3-图像超分辨率方法 | |
| 实  验  内  容 | 1. 实现一种图像超分辨率方法在set5数据集上的测试，得到超分辨率图像，测量与其原始真实图像之间的PSNR、SSIM指标值。对所选择方法的细节进行介绍，并试着讨论该方法可能存在的优缺点，以及可能的改进方向。 |
| 实  验  思  路 | 1. 搭建ESPCN网络。 2. 下载set5数据集，使用mindspore去加载测试数据。 3. 设计测试过程。开始测试，保存结果。 |
| 实  验  步  骤 | 该代码执行以下步骤：   1. 导入所需的库：mindspore、mindspore.nn、torch、skimage.metrics、PIL、math、numpy和cv2。 2. 使用MindSpore库中的nn.Cell类定义ESPCN（Enhanced Super-Resolution Generative Adversarial Network）模型。该模型由两个部分组成：first\_part和last\_part，它们使用nn.SequentialCell定义为顺序层。 3. 定义两个实用函数：do\_Convert\_ycbcr\_to\_rgb和do\_Convert\_rgb\_to\_ycbcr，用于在RGB和YCbCr颜色空间之间转换图像。 4. 定义一个预处理函数do\_Preprocess，它接受图像作为输入，将其转换为YCbCr颜色空间，对Y通道值进行归一化，并准备将其输入ESPCN模型。 5. 定义一个函数do\_Bicubic，它接受一个测试图像和一个缩放因子，执行双三次插值来对图像进行降采样，得到一个低分辨率图像（lr\_img），并返回原始图像（src\_img）和双三次上采样图像（bi\_img），以供后续比较。 6. 定义一个函数do\_ESPCN，它接受预训练的ESPCN模型的路径、低分辨率图像（lr\_img）和YCbCr图像（ycbcr）。它加载模型，将其设置为评估模式，通过模型处理低分辨率图像以获得超分辨率图像（preds），将图像转换回RGB颜色空间，剪辑像素值，并返回输出图像。 7. 定义一个函数do\_Calculate\_Metrics，它接受源图像和超分辨率图像的路径，加载图像，将它们转换为灰度图像，并计算两个图像之间的PSNR（峰值信噪比）和SSIM（结构相似性指数）指标。 8. 在if name == "\_\_main\_\_":块中： 9. 将路径变量设置为空字符串（假设图像和预训练模型与脚本位于同一目录中）。使用PIL.Image.open打开测试图像。 10. 对测试图像执行双三次插值，以获得源图像（src\_img）、低分辨率图像（lr\_img）和双三次上采样图像（bi\_img）。 11. 保存源图像和双三次上采样图像。 12. 使用do\_Preprocess对低分辨率图像进行预处理。 13. 调用do\_ESPCN函数获取超分辨率图像（output\_img）。 14. 保存超分辨率图像。 15. 调用do\_Calculate\_Metrics函数计算源图像和超分辨率图像之间的PSNR和SSIM指标。 16. 打印计算得到的PSNR和SSIM值。   总体步骤涉及加载测试图像，执行双三次插值以获得低分辨率图像，对低分辨率图像进行预处理，将ESPCN模型应用于低分辨率图像，计算源图像和超分辨率图像之间的指标，并保存结果。 |
| 实  验  结  果 | |  |  |  |  | | --- | --- | --- | --- | | 原图像 | 下采样后 | 网络输出 | 评价 | |  |  |  | PSNR:34.25  SSIM: 0.92 | |  |  |  | PSNR:33.29  SSIM: 0.93 | |  |  |  | PSNR:26.26  SSIM: 0.83 | |  |  |  | PSNR:33.55  SSIM: 0.84 | |  |  |  | PSNR:29.45  SSIM: 0.89 |   该方法的优点和改进方向如下：  优点：  使用了ESPCN模型，该模型在超分辨率任务上表现出良好的性能。  通过双三次插值将图像降采样为低分辨率图像，再使用ESPCN模型进行超分辨率重建，可以提高图像的细节和清晰度。  使用了PSNR和SSIM这两个常用的图像质量评价指标来衡量超分辨率重建的效果，这些指标可以提供量化的评估结果。  改进方向：  使用更先进的超分辨率模型：当前代码使用的是ESPCN模型，可以尝试使用更先进的模型，如EDSR、SRGAN等，以进一步提升超分辨率重建的质量。  数据增强和训练策略：可以尝试使用数据增强技术，如随机裁剪、旋转、翻转等，增加训练样本的多样性。此外，调整训练策略，如学习率衰减、迭代次数等，可以提升模型的性能。  引入感知损失：除了使用传统的均方误差损失函数，可以考虑引入感知损失函数，如VGG网络提取的特征损失，以增强对图像细节和结构的重建能力。  数据集的多样性：使用更大规模、多样性的训练数据集，可以提升模型的泛化能力，使其适应更广泛的图像场景。  超参数调优：通过系统地调整超参数（如模型的层数、通道数、学习率等），可以找到更适合任务的模型配置，提高性能。 |